Second-Order Type Isomorphisms Through Game Semantics

Joachim de Lataillade

Second-order Type Isomorphisms through Game Semantics

Joachim de Lataillade

Équipe Preuves, Programmes et Systèmes
CNRS – Université Paris 7 Denis Diderot

Abstract

The characterization of second-order type isomorphisms is a purely syntactical problem that we propose to study under the enlightenment of game semantics. We study this question in the case of second-order $\lambda\mu$-calculus, which can be seen as an extension of system F to classical logic, and for which we define a categorical framework: control hyperdoctrines.

Our game model of $\lambda\mu$-calculus is based on polymorphic arenas (closely related to Hughes' hyperforests) which evolve during the play (following the ideas of Murawski-Ong). We show that type isomorphisms coincide with the "equality" on arenas associated with types. Finally we deduce the equational characterization of type isomorphisms from this equality. We also recover from the same model Roberto Di Cosmo's characterization of type isomorphisms for system F.

This approach leads to a geometrical comprehension on the question of second order type isomorphisms, which can be easily extended to some other polymorphic calculi including additional programming features.

Key words: Types Isomorphisms, Second-order $\lambda\mu$-calculus, Game Semantics, Hyperdoctrines, Control Categories

1 Introduction

Denotational semantics Defining a semantic for a language is a fundamental tool for understanding the way this language works. Thus, semantics is a very active domain of research in theoretical computer science: in particular, there has been an important investigation on semantics which could modelize a language as precisely as possible; this has led to the emergence of game semantics in the early 90s, whose success is due to the deep adequation of its models with the syntax. The present work illustrates the ability of game semantics to modelize a language precisely: consequently, it is possible to extract from the model some properties of the language. So, this work has to be understood as an example of accomplishment of the original goal of denotational semantics: using abstract tools to prove concrete properties on a programming language. In this article, the property we extract concerns a non-trivial problem, the characterization of type isomorphisms for second-order languages.

Type isomorphisms. The problem of type isomorphisms is a syntactical question: two types A and B are isomorphic ($A \simeq B$) if there exist two terms $f : A \to B$ and $g : B \to A$ such that $f \circ g = id_B$ and $g \circ f = id_A$. This equivalence relation on data types allows to translate a program from one type to the other without any change on the calculatory meaning of the program. Thus, a search in a library up to type isomorphism will help the programmer to find all the functions that can potentially serve his purpose, and to reuse them in the new typing context [Rit91]. This is particularly appealing with functional languages, because in this case the type can really be seen as a partial specification of the program: such a library search up to isomorphisms has been implemented in particular for Caml Light by Jérôme Vouillon. It can also be used in proof assistants to help finding proofs in libraries and reusing them [BP01] (for more details on the use of type isomorphisms in computer science, see [DC95]).

When dealing with type isomorphisms, the key problem, given a programming language, is to find a characterization of isomorphic types through an equational system. This can be done either syntactically (by working directly on terms) or semantically (by using an adequate model of the calculus, i.e. such that there are no more isomorphisms in the model than in the language). For the λ-calculus, the problem has been solved semantically as early as in 1981 [Sol83], but Olivier Laurent has recently proposed a new approach based on game semantics [Lau05]: taking the usual HON game model for λ-calculus (which we call the *propositional game model*), he proved that the equality modulo isomorphism in the syntax corresponds to the notion of equality between forests, and proved the equational characterization of isomorphisms by this means. The main steps of his proof are summed up

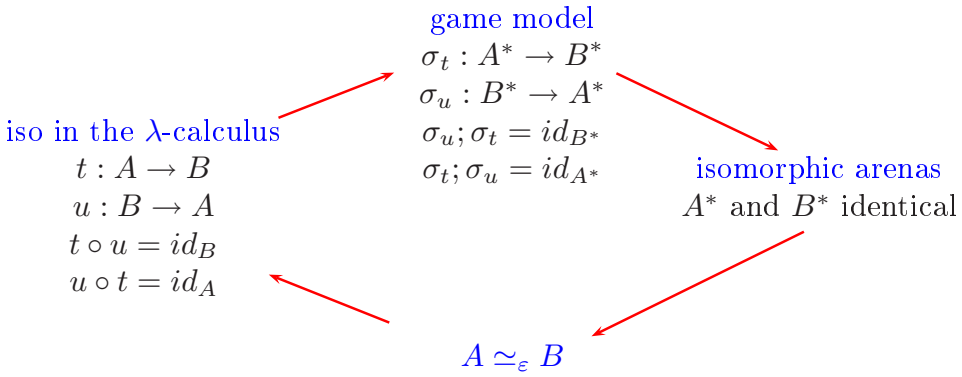


Fig. 1. Steps of the proof of Olivier Laurent in his work on type isomorphisms

on figure 1. The advantage of this point of view is that it immediately gave him a characterization of type isomorphisms for the $\lambda\mu$-calculus, requiring no additional work.

$\lambda\mu$2-calculus. The calculus we consider in this work is the call-by-name disjunctive second-order $\lambda\mu$-calculus (shortly, $\lambda\mu$2). The $\lambda\mu$-calculus has been introduced by Parigot [Par92] as a way to add the notion of *control* to λ-calculus, and hence to associate a calculus to classical logic. There are a call-by-value version and a call-by-name version of this calculus, that Peter Selinger proved to be isomorphic one to the other [Sel01]. The $\lambda\mu$2-calculus is just an extension of this calculus to second order: here we will consider a Church-style presentation of second-order terms.

As far as we know, the characterization of type isomorphisms for $\lambda\mu$2 has not been done yet. However, using the results of Roberto Di Cosmo concerning system F [DC95] and of Olivier Laurent concerning the $\lambda\mu$-calculus [Lau05], one can suggest that the equational system that characterizes type isomorphisms for $\lambda\mu$2 is the system $\simeq_\varepsilon$ presented on figure 2 (note that we have now an equality corresponding the interaction between $\forall$ and $\parr$). We propose in this paper a semantic demonstration of this result, in the spirit of the work of Olivier Laurent.

Categorical models for $\lambda\mu$2. The first part of this work is dedicated to the description of a categorical structure which generates models of $\lambda\mu$2. This construction is essentially a mix between the structure of hyperdoctrines, introduced by Lawvere [Law70], which have been proved to be a categorical model of system F, and the control categories, invented by Peter Selinger [Sel01] to give a categorical characterization of models of the $\lambda\mu$-calculus. The only points that require more caution are at the interface between the two structures, i.e. at the interaction between the functor Π_I that models quantification in a hyperdoctrine and the binoidal functor $\parr$ of control categories.

Game semantics. Models of second order calculi do not come about easily

$$A \times B \simeq_\varepsilon B \times A \qquad A \times \top \simeq_\varepsilon A \qquad \forall X.\forall Y.A \simeq_\varepsilon \forall Y.\forall X.A$$
$$A \times (B \times C) \simeq_\varepsilon (A \times B) \times C \qquad \forall X.\top \simeq_\varepsilon \top \qquad \forall X.(A \times B) \simeq_\varepsilon \forall X.A \times \forall X.B$$
$$A \to (B \to C) \simeq_\varepsilon (A \times B) \to C \qquad \top \to A \simeq_\varepsilon A \qquad A \parr B \simeq_\varepsilon B \parr A$$
$$(A \to B) \parr C \simeq_\varepsilon A \to (B \parr C) \qquad A \to \top \simeq_\varepsilon \top \qquad A \parr (B \parr C) \simeq_\varepsilon (A \parr B) \parr C$$
$$(A \times B) \parr C \simeq_\varepsilon (A \parr C) \times (B \parr C) \qquad \top \parr A \simeq_\varepsilon \top$$
$$\bot \parr A \simeq_\varepsilon A$$
$$A \parr \forall X.B \simeq_\varepsilon \forall X.(A \parr B) \qquad \text{if } X \text{ does not appear free in } A$$

Fig. 2. Equational system for type isomorphisms in $\lambda\mu 2$

due to impredicativity. Among the different possibilities, we choose models based on game semantics because of their high degree of adequation with the syntax: indeed, game semantics has been widely used to construct fully complete models for various calculi, such as PCF [AJM00, HO00], μPCF [Lai97], Idealized Algol [AM99], etc. This means that this semantics gives a very faithful description of the behavior of the syntax modulo reduction rules in the system. And this is precisely what we need to deal semantically with type isomorphisms: a model which is so precise that it contains no more isomorphisms than the syntax.

The first game model of system F was a complete HON-style game model by Hughes [Hug00] from which we inherit the notion of hyperforests (i.e. forests with more structure); unfortunately the complex mechanism for interaction in this model prevents us from calculating isomorphisms efficiently. Murawski and Ong developed an alternative model (for affine polymorphism) based on the notion of evolving games [MO01]: we will reuse this idea in the context of a HON-style game. Finally, Abramsky and Jagadeesan built a model dedicated to generic polymorphism [AJ03], and thus their model is not appropriate for our objectives.

The model. The second part of this paper presents polymorphic arenas and strategies on these arenas: polymorphic arenas are forests with a precise structure for nodes that make them very closed to second-order formulas. A structure of hyperforest can be extracted from these arenas (however, note that hyperforests are not *the* basic structure used to define arenas). The notion of move in a polymorphic arena is more sophisticated than in propositional game semantics, but these moves carry all the second-order structure, so that the definitions of plays, views, strategies, etc, will not change.

We prove that we have obtained a model for $\lambda\mu 2$ by using the tools defined in

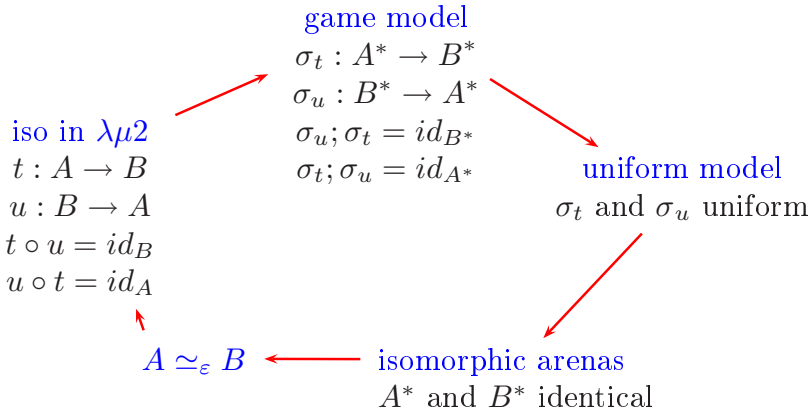


Fig. 3. Steps of our demonstration

the first part. In this model, the two players **O** and **P** have a very symmetrical behavior, so that interaction is easy to define. But this symmetry is paid by the fact that this model, being very liberal, is far from being complete (which is not a problem by itself in our perspective), and in particular it has too many isomorphisms compared to our language.

Uniformity. That is why we add a new property for strategies, uniformity (also inspired partly by [MO01]), which breaks this symmetry between players and gives raise to a sub-model (which is also far from being complete, but we do not care for that) where the isomorphisms will happen to have exactly the same form as in $\lambda\mu 2$: uniformity is just an ad hoc property, precisely defined to retrieve exactly $\lambda\mu 2$ isomorphisms.

The core theorem of our work on isomorphisms consists in proving that, in the uniform model, the existence of a game isomorphism between two polymorphic arenas ($A \simeq_g B$) induces that these two arenas are equal in the most natural sense ($A \simeq_a B$). Then we can conclude on the characterization of type isomorphisms: if we denote A^* the interpretation of a type A in the uniform model, then we have:

$$A \simeq B \Leftrightarrow A^* \simeq_g B^* \Leftrightarrow A^* \simeq_a B^* \Leftrightarrow A \simeq_\epsilon B$$

The main steps of this reasoning are summed up on figure 3. As an easy corollary of this result, one is able to retrieve the characterization of type isomorphisms for Church-style system F, proved syntactically by Roberto Di Cosmo [DC95]. Moreover, the results can also be extended easily to some little extensions of the calculus, like a calculus with a fixpoint operator. Finally, the geometrical aspect of this work leads us to an interesting remark: hyperforests, which naturally carry the equivalence corresponding to type isomorphisms, happen to be a very significant description of second-order formulas.

2 Control hyperdoctrines

2.1 The second-order λμ-calculus

The Curry-Howard correspondence, illustrated for intuitionistic logic by the simply typed λ-calculus, can be extended to classical logic through Michel Parigot's $\lambda\mu$-calculus [Par92]. It adds new operators to the λ-calculus, in order to enable the notion of *control*. Hence, the calculus allows to use the output as if it was sent to many outputs, which correspond to the sequents with several conclusions of classical logic. As an example, the well-known control command `call/cc` and its semantics can be encoded in the $\lambda\mu$-calculus. There are two different paradigms, which differ in the reduction rules of the control operators: the call-by-name and the call-by-value $\lambda\mu$-calculi. Peter Selinger proved in [Sel01] that these two calculi are dual.

Here we consider the second-order extension of this calculus, in a call-by-name paradigm, and with the disjunction type introduced by Selinger in [Sel01]. This system will be called $\lambda\mu 2$ in the rest of the paper.

The grammar of types is the following:

$$A = \top \mid \bot \mid X \mid A \times A \mid A \to A \mid A \parr A \mid \forall X.A$$

The grammar of terms is:

$$\begin{aligned} t ::= {} & x \mid \star \mid (t,t) \mid \pi_1(t) \mid \pi_2(t) \mid tt \mid \lambda x^A.t \mid [\alpha]t \mid \mu\alpha^A.t \\ & \mid [\alpha,\beta]t \mid \mu(\alpha^A,\beta^B).t \mid \Lambda X.t \mid t\{A\} \end{aligned}$$

The variables α will be called **names**. If $[\alpha]t$ appears in the scope of a $\mu\alpha^A$ it will be called a **bound name**; if not it is a **free name**; the set of free names of a term t will be denoted by $FN(t)$. The set of free term variables (resp. free type variables) appearing in a term is denoted $FV(t)$ (resp. $FTV(t)$).

In order to control the free type variables appearing in a sequent, we introduce the enabling judgement $\vec{X} \Vdash A$: it expresses the fact that the free type variables of a type A are chosen among $X_1, \ldots, X_n$, and it is defined by the following inference rules:

$$\frac{X \in \vec{X}}{\vec{X} \Vdash X} \qquad \frac{}{\vec{X} \Vdash \top} \qquad \frac{}{\vec{X} \Vdash \bot}$$

$$\frac{\vec{X} \Vdash A \qquad \vec{X} \Vdash B}{\vec{X} \Vdash A \to B} \qquad \frac{\vec{X} \Vdash A \qquad \vec{X} \Vdash B}{\vec{X} \Vdash A \times B} \qquad \frac{\vec{X} \Vdash A \qquad \vec{X} \Vdash B}{\vec{X} \Vdash A \parr B}$$

$$\frac{\vec{X}, X \Vdash A}{\vec{X} \Vdash \forall X.A}$$

The sequents of our calculus take the form $\vec{X}; \Gamma \vdash t : A \mid \Delta$ where t is the term, A is the type, Γ is a context for variables (a sequence of typing assignments $x_i : A_i$, where x_i is a variable that appears at most once in Γ), Δ is a context for names (a set of typing assignments $\alpha_i : A_i$, where α_i is a name that appears at most once in Δ) and $\vec{X}$ is a set of type variables. The typing rules are:

$$\text{(ax)}\ \frac{\vec{X} \Vdash A_1 \quad \dots \quad \vec{X} \Vdash A_n \quad \vec{X} \Vdash B_1 \quad \dots \quad \vec{X} \Vdash B_p}{\vec{X}; x_1 : A_1, \dots, x_n : A_n \vdash x_i : A_i \mid \alpha_1 : B_1, \dots, \alpha_p : B_p}$$

$$(\top)\ \frac{\vec{X} \Vdash \Gamma \qquad \vec{X} \Vdash \Delta}{\vec{X}; \Gamma \vdash \star : \top \mid \Delta}$$

$$(\to I)\ \frac{\vec{X}; \Gamma, x : A \vdash t : B \mid \Delta}{\vec{X}; \Gamma \vdash \lambda x^A.t : A \to B \mid \Delta}$$

$$(\to E)\ \frac{\vec{X}; \Gamma \vdash t : A \to B \mid \Delta \qquad \vec{X}; \Gamma \vdash u : A \mid \Delta}{\vec{X}; \Gamma \vdash tu : B \mid \Delta}$$

$$(\times I)\ \frac{\vec{X}; \Gamma \vdash t : A \mid \Delta \qquad \vec{X}; \Gamma \vdash u : B \mid \Delta}{\vec{X}; \Gamma \vdash (t, u) : A \times B \mid \Delta}$$

$$(\times E1)\ \frac{\vec{X}; \Gamma \vdash t : A \times B \mid \Delta}{\vec{X}; \Gamma \vdash \pi_1(t) : A \mid \Delta} \qquad (\times E2)\ \frac{\vec{X}; \Gamma \vdash t : A \times B \mid \Delta}{\vec{X}; \Gamma \vdash \pi_2(t) : B \mid \Delta}$$

$$\text{(naming rule)}\ \frac{\vec{X}; \Gamma \vdash t : A \mid \Delta}{\vec{X}; \Gamma \vdash [\alpha]t : \bot \mid \Delta}\ \text{if } \alpha : A \in \Delta$$

$$(\mu\text{-rule})\ \frac{\vec{X}; \Gamma \vdash t : \bot \mid \alpha : A, \Delta}{\vec{X}; \Gamma \vdash \mu\alpha^A.t : A \mid \Delta}$$

$$\text{(double naming rule)}\ \frac{\vec{X}; \Gamma \vdash t : A \parr B \mid \Delta}{\vec{X}; \Gamma \vdash [\alpha, \beta]t : \bot \mid \Delta}\ \text{if } \alpha : A, \beta : B \in \Delta$$

$$(\text{double } \mu\text{-rule})\ \frac{\vec{X}; \Gamma \vdash t : \bot \mid \alpha : A, \beta : B, \Delta}{\vec{X}; \Gamma \vdash \mu(\alpha^A, \beta^B).t : A \parr B \mid \Delta}$$

$$(\forall I)\ \frac{\vec{X}, X; \Gamma \vdash t : A \mid \Delta}{\vec{X}; \Gamma \vdash \Lambda X.t : \forall X.A \mid \Delta}\ \text{if } X \notin FTV(\Gamma) \cup FTV(\Delta)$$

$$(\forall E)\ \frac{\vec{X}; \Gamma \vdash t : \forall X.A \mid \Delta \qquad \vec{X} \Vdash B}{\vec{X}; \Gamma \vdash t\{B\} : A[B/X] \mid \Delta}$$

Finally, the equational theory of $\lambda\mu 2$ is defined by the sequents $\vec{X}; \Gamma \vdash t = u : A \mid \Delta$ (with $\vec{X}; \Gamma \vdash t : A \mid \Delta$ and $\vec{X}; \Gamma \vdash u : A \mid \Delta$) generated by congruence relations that can be classified as follows:

λ-calculus with products:

$$\begin{array}{rlll}
t = \star & : \top & & ((\top)) \\
\pi_1((u,v)) = u & : A & & ((\pi_1)) \\
\pi_2((u,v)) = v & : B & & (\pi_2) \\
(\pi_1(u), \pi_2(u)) = u & : A \times B & & (\times) \\
(\lambda x^A.t)u = t[u/x] & : B & & (\beta) \\
\lambda x^A.tx = t & : A \to B & \text{if } x \notin FV(t) & (\eta)
\end{array}$$

$\lambda\mu$-calculus with disjunction:

$$\begin{array}{rlll}
(\mu\alpha^{A\to B}.t)u = \mu\beta^B.t[[\beta](-)u/[\alpha](-)] & : B & \text{if } \beta \notin FN(t,u) & (\mu^{\to}) \\
\pi_i(\mu\alpha^{A_1\times A_2}.t) = \mu\beta^{A_i}.t[[\beta]\pi_i(-)/[\alpha](-)] & : A_i & \text{if } \beta \notin FN(t) & (\mu^{\times}) \\
[\beta,\gamma](\mu\alpha^{A\parr B}.t) = t[[\beta,\gamma](-)/[\alpha](-)] & : \bot & & (\mu^{\parr}) \\
(\mu\alpha^{\forall X.A}.t)\{B\} = \mu\beta^{A[B/X]}.t[[\beta](-)\{B\}/[\alpha](-)] & : A[B/X] & \text{if } \beta \notin FN(t) & (\mu^{\forall})
\end{array}$$

$$\begin{array}{rlll}
[\alpha']\mu\alpha^A.t = t[\alpha'/\alpha] & : \bot & & (\rho^{\mu}) \\
[\alpha',\beta']\mu(\alpha^A,\beta^B).t = t[\alpha'/\alpha,\beta'/\beta] & : \bot & & (\rho^{\parr}) \\
[\xi]t = t & : \bot & \text{if } \xi : \bot \in \Delta & (\rho^{\bot})
\end{array}$$

$$\begin{array}{rlll}
\mu\alpha^A.[\alpha]t = t & : A & \text{if } \alpha \notin FN(t) & (\theta^{\mu}) \\
\mu(\alpha^A,\beta^B).[\alpha,\beta]t = t & : A \parr B & \text{if } \alpha,\beta \notin FN(t) & (\theta^{\parr})
\end{array}$$

Second order quantification:

$$\begin{array}{rlll}
(\Lambda X.t)\{B\} = t[B/X] & : A[B/X] & & (\beta 2) \\
\Lambda X.t\{X\} = t & : \forall X.A & \text{if } X \notin FTV(t) & (\eta 2)
\end{array}$$

In the above relations, the contextual substitution $s_{\alpha,C}(M) = M[C(-)/[\alpha](-)]$ where M is a term, $t \mapsto C(t)$ is an operation on terms and $\alpha : A$ appears in the name context, has to be defined by induction on M:

- $s_{\alpha,C}([\alpha]M) = C([\alpha]s_{\alpha,C}(M))$
- $s_{\alpha,C}([\alpha,\beta]M) = C(\mu\alpha^A.[\alpha,\beta]s_{\alpha,C}(M))$
- $s_{\alpha,C}([\beta,\alpha]M) = C(\mu\alpha^A.[\beta,\alpha]s_{\alpha,C}(M))$
- $s_{\alpha,C}$ commutes with all other base operations on terms (with the requirement to avoid captures).

Now that the system $\lambda\mu 2$ is completely defined, one can give the definition of a type isomorphism:

Definition 1 (type isomorphism) *Let A and B be two types of $\lambda\mu 2$. We say that there is a type isomorphism between A and B if there exist two terms t and u such that:*

- $\vec{X};\vdash t : A \to B \mid$
- $\vec{X};\vdash u : B \to A \mid$
- $\lambda x^B.t(ux) = \lambda x^B.x$
- $\lambda y^A.u(ty) = \lambda y^A.y$

2.2 Definition of a control hyperdoctrine

We wish to give a categorical model of $\lambda\mu 2$. For this we use two ingredients : first, the notion of *hyperdoctrine*, introduced by Lawvere [Law70], with which Seely [See87] and Pitts [Pit88] have proposed a categorical interpretation of system F; second, the notion of *control category* [Sel01], which introduces a disjunction $\parr$ to characterize models of the $\lambda\mu$-calculus. We chose to give preference to control categories rather than categories of continuations, because using continuation categories would require to build a CPS-translation transforming the connector $\forall$ into the connector $\exists$, and to build a theory for categories of continuations with the connector $\exists$: as our model is based on the interpretation of the connector $\forall$, we did not choose this option.

In the following definition, **CCC** is the category of cartesian closed categories with strict morphisms of ccc's ($G : C \to D$ is a strict morphism if the specified cartesian closed structure of **C** is sent to the specified cartesian closed structure of **D**).

Definition 2 (hyperdoctrine) *An **hyperdoctrine** **H** is specified by:*

- *a base category $|\mathbf{H}|$ with terminal object $\top$ and binary products*

- *a distinguished object U in $|\mathbf{H}|$ such that for all $I \in |\mathbf{H}|$ there exists $n \in \mathbb{N}$ such that $I = U^n$ (with the convention $U^0 = \top$); we denote $\pi_n^i : U^n \to U$ the projection on the ith component, and $\pi_{U^n} = \pi_{n+1}^1 \times \ldots \times \pi_{n+1}^n : U^{n+1} \to U^n$*
- *a functor $F : |\mathbf{H}|^{op} \to \mathbf{CCC}$ such that if we compose F with the forgetful functor $fff : \mathbf{CCC} \to \mathbf{Set}$ we obtain the functor $|\mathbf{H}|(-, U)$*
- *for each $I \in |\mathbf{H}|$, a functor $\Pi_I : F(I \times U) \to F(I)$ such that :*
  - *Π_I is right adjoint to the functor $F(\pi_{I\times U}) : F(I) \to F(I \times U)$*
  - *Π_I is natural in I: for any $\alpha : I \to J$, $F(\alpha) \circ \Pi_J = \Pi_I \circ F(\alpha \times id_U)$*
  - *for any $\alpha : I \to J$, for any object A of $F(J \times U)$, the morphism $(F(\alpha) \circ \Pi_J)(A) \to (\Pi_I \circ F(\alpha \times id_U))(A)$ generated by the adjunction is the identity.*

*The functors $F(C)$, with C object of $|\mathbf{H}|^{op}$, are called the **specialization functors**.*

The intuitions of such a categorical description are the following: the objects (resp. the morphisms) of $F(U^n)$ will correspond to the types (resp. the terms) where each free type variable that appears is chosen between $X_1, \ldots, X_n$. Hence, $F(\pi_{I\times U}) : F(I) \to F(I \times U)$ is simply the standard way to make the number of free type variables growing: thus we will often address this functor as $- \mapsto (-)_I^{I\times U}$. Π_I makes this number of free type variables decreasing by quantifying one of them.

We adopt the following notations for a cartesian closed category: $\diamond_A : A \to 1$ is the terminal arrow, π_1, π_2 are the two projections, (f, g) is for pairing, $\epsilon_{A,B} : B^A \times A \to B$ is the evaluation and the curryfication of a map $f : B \times A \to C$ is denoted $\Lambda(f) : B \to C^A$. We sometimes note $\xrightarrow{ccc}$ to denote trivial isomorphisms in a ccc. We also note $\kappa : Hom_{F(I\times U)}((C)_I^{I\times U}, A) \to Hom_{F(I)}(C, \Pi_I(A))$ the bijection associated with the adjunction $F(\pi_{I\times U}) \dashv \Pi_I$.

We now introduce the notion of **control hyperdoctrine**, in order to adapt hyperdoctrines to a description of second-order *classical* logic: the path from hyperdoctrines to control hyperdoctrines will take the same form as the one from cartesian closed categories to control categories:

- we first give the hyperdoctrine structure
- we introduce the symmetric pretensor $\parr$ together with the neutral element $\bot$
- we then require the existence of codiagonals, i.e. for each object A two central morphisms $i_A : \bot \to A$ and $\Delta_A : A \parr A \to A$ such that $\langle A, i_A, \Delta_A \rangle$ is a symmetric monoid compatible with the premonoidal structure
- we introduce a new condition, **hypermonoidality**, that asks for the commutation of the specialization functors with the premonoidal structure and the codiagonals, and the preservation of centrality through the hyperdoctrine adjunction
- we require the distributivity of $\parr$ over the cartesian product
- we introduce the **exponential strength**: the (already existing) mor-

phism $s_{A,B,C} : B^A \parr C \to (B \parr C)^A$ is a *natural isomorphism* which respects some coherence conditions
- we also introduce the **quantification strength**: the (already existing) morphism $p_{A,B} : \Pi_I(A) \parr B \to \Pi_I(A \parr (B)_I^{I \times U})$ is a *natural isomorphism* which respects a condition of centrality.

In the following definition, $[C]$ denotes the class of objects of a category C, regarded as a discrete subcategory.

Definition 3 (binoidal hyperdoctrine) *A **binoidal hyperdoctrine** $\mathbf{H}$ is an hyperdoctrine together with, for each $I \in |\mathbf{H}|$, a binoidal functor $\parr_I$, i.e. a couple of two bifunctors $\parr_I^1 : F(I) \times [F(I)] \to F(I)$ and $\parr_I^2 : [F(I)] \times F(I) \to F(I)$ such that $A \parr_I^1 B = B \parr_I^2 A$ for all pairs of objects A, B.*

We recall the definition of a central morphism: in the ccc $F(I)$, $f : A \to A'$ is central if for every $g : B \to B'$ one has $(f \parr_I B') \circ (A \parr_I g) = (A' \parr_I g) \circ (f \parr_I B)$ and $(B' \parr_I f) \circ (g \parr_I A) = (g \parr_I A') \circ (B \parr_I f)$.

Definition 4 (premonoidal hyperdoctrine) *A **premonoidal hyperdoctrine** is a binoidal hyperdoctrine $\mathbf{H}$ together with, for each $I \in |\mathbf{H}|$, an object $\bot_I$ and central natural isomorphisms $a_{A,B,C} : (A \parr_I B) \parr_I C \to A \parr_I (B \parr_I C)$, $l_A : A \to A \parr_I \bot_I$ and $r_A : A \to \bot_I \parr_I A$ making the following diagrams commute:*

$$\begin{array}{ccccc}
((A \parr_I B) \parr_I C) \parr_I D & \xrightarrow{a} & (A \parr_I B) \parr_I (C \parr_I D) & \xrightarrow{a} & A \parr_I (B \parr_I (C \parr_I D)) \\
\downarrow{\scriptstyle a \parr D} & & & & \uparrow{\scriptstyle A \parr a} \\
(A \parr_I (B \parr_I C)) \parr_I D & & \xrightarrow{\qquad a \qquad} & & A \parr_I ((B \parr_I C) \parr_I D)
\end{array}$$

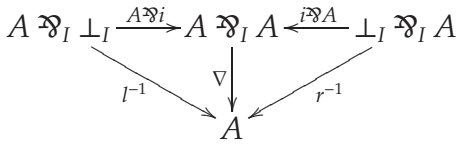
$$\begin{array}{ccc}
 & A \parr_I B & \\
{\scriptstyle l \parr B}\swarrow & & \searrow{\scriptstyle A \parr r} \\
(A \parr_I \bot_I) \parr_I B & \xrightarrow{a} & A \parr_I (\bot_I \parr_I B)
\end{array}$$

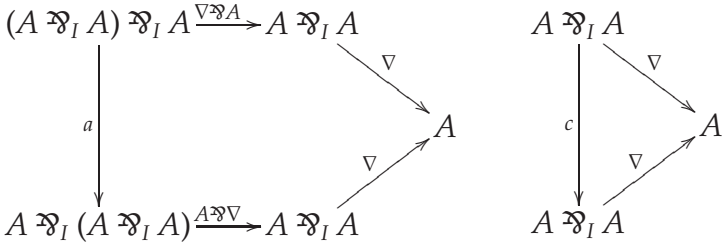
*It is called a **symmetric premonoidal hyperdoctrine** if there are in addition central natural isomorphisms $c_{A,B} : A \parr_I B \to B \parr_I A$ such that $c_{A,B} \circ c_{B,A} = id_{A \parr_I B}$ and:*

$$\begin{array}{ccccc}
(A \parr_I B) \parr_I C & \xrightarrow{a} & A \parr_I (B \parr_I C) & \xrightarrow{c} & (B \parr_I C) \parr_I A \\
\downarrow{\scriptstyle c \parr C} & & & & \downarrow{\scriptstyle a} \\
(B \parr_I A) \parr_I C & \xrightarrow{a} & B \parr_I (A \parr_I C) & \xrightarrow{B \parr c} & B \parr_I (C \parr_I A)
\end{array}$$

$$\begin{array}{ccc}
 & A & \\
{\scriptstyle l}\swarrow & & \searrow{\scriptstyle r} \\
A \parr_I \bot_I & \xrightarrow{c} & \bot_I \parr_I A
\end{array}$$

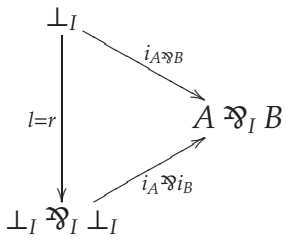
Definition 5 (symmetric monoid, codiagonals) *Let* $\mathbf{H}$ *be a symmetric premonoidal hyperdoctrine. A symmetric monoid in* $\mathbf{H}$ *for an object* $A \in F(I)$ *(*$I \in |\mathbf{H}|$*) is a pair of central morphisms* $i_A : \bot_I \to A$ *and* $\nabla_A : A \parr_I A \to A$ *such that:*

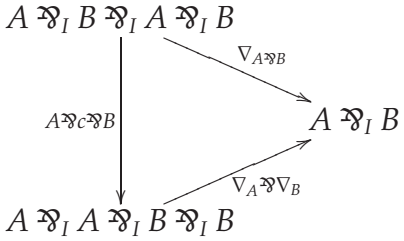




We say that a a symmetric premonoidal hyperdoctrine has ***codiagonals*** *if, for each* $I \in |\mathbf{H}|$*, there is a symmetric monoid for every* $A \in F(I)$*, which is compatible with the premonoidal structure:*

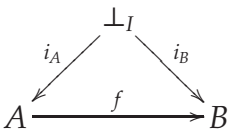
$i_{\bot_I} = id_{\bot_I}$

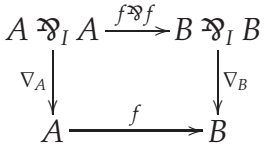




The central morphism ∇_A recovers the notion of *contraction* from linear logic. One can also define the *weakening* in a premonoidal hyperdoctrine with codiagonals: $w = A \xrightarrow{l} A \parr_I \bot_I \xrightarrow{A \parr i} A \parr_I B$.

Definition 6 (focality) *A morphism* $f : A \to B$ *is* ***focal*** *if it is central and the two following diagrams commute:*





A premonoidal *category* with codiagonals will be called a pre-control category. A strict morphism $\mu : C \to D$ of pre-control categories is such that it

sends each element of the structure of C in the corresponding element of the structure D: $\mu(A \parr B) = \mu(A) \parr \mu(B)$, $\mu(\bot) = \bot$, $\mu(\sigma \parr A) = \mu(\sigma) \parr \mu(A)$, $\mu(a_{A,B,C}) = a_{\mu(A),\mu(B),\mu(C)}$, etc.

Definition 7 (hypermonoidality) *Let* **H** *be a symmetric premonoidal hyperdoctrine with codiagonals. We say that* **H** *has* ***hypermonoidality*** *if the specialization functors are strict morphisms of pre-control categories and if κ, κ^{-1} preserve centrality of morphisms.*

Definition 8 (distributivity) *Let* **H** *be a symmetric premonoidal hyperdoctrine with codiagonals.* **H** *is said to be* ***distributive*** *if:*

- *the projections π_1 and π_2 are focal*
- *for each $I \in |\mathbf{H}|$ and $A \in F(I)$ the functor $- \parr_I A$ preserves finite products: the natural morphisms $(\pi_1 \parr_I C, \pi_2 \parr_I C) : (A \times B) \parr_I C \to (A \parr_I C) \times (B \parr_I C)$ and $\diamond_{1 \parr_I C} : 1 \parr_I C \to 1$ are isomorphisms, whose inverses are respectly denoted $d_{A,B,C}$ and $\diamond'_C$.*

Definition 9 (control hyperdoctrine) *Let* **H** *be a distributive symmetric premonoidal hyperdoctrine with codiagonals and hypercentrality. For $A, B, C \in F(I)$, let $s_{A,B,C} : (B^A \parr_I C) \to (B \parr_I C)^A$ be the canonical morphism obtained by currying*

$$\hat{\epsilon}_{A,B,C} : (B^A \parr_I C) \times A \xrightarrow{(B^A \parr C) \times (l; A \parr i)} (B^A \parr_I C) \times (A \parr_I C) \xrightarrow{d} (B^A \times A) \parr_I C \xrightarrow{\epsilon \parr C} B \parr_I C$$

For $A \in F(I \times U)$ and $B \in F(I)$, let

$$p_{A,B} = \kappa(\kappa^{-1}(id_{\prod_I(A)}) \parr_{I \times U} (B)_I^{I \times U}) : \prod_I(A) \parr_I B \to \prod_I(A \parr_{I \times U} (B)_I^{I \times U})$$

H *is called a* ***control hyperdoctrine*** *if*

- *$s_{A,B,C}$ is a natural isomorphism in C satisfying:*

$$\begin{array}{ccc} B^A \parr_I C^D & \xrightarrow{\quad s' \quad} & (B^A \parr_I C)^D \\ \downarrow{\scriptstyle s} & & \downarrow{\scriptstyle s^D} \\ (B \parr_I C^D)^A & \xrightarrow{s'^A} ((B \parr_I C)^D)^A \xrightarrow{ccc} & ((B \parr_I C)^A)^D \end{array}$$

  where $s'_{A,B,C} = B \parr_I C^A \xrightarrow{c} C^A \parr_I B \xrightarrow{s} (C \parr_I B)^A \xrightarrow{c^A} (B \parr_I C)^A$ and:

$$\begin{array}{ccccc} B^A \parr_I B^A & \xrightarrow{s'} & (B^A \parr_I B)^A & \xrightarrow{s^A} & (B \parr_I B)^{A \times A} \\ & {\scriptstyle \nabla_{B^A}} \searrow & & \swarrow {\scriptstyle \nabla_A^{\Delta_A}} & \\ & & B^A & & \end{array} \qquad \begin{array}{ccc} \bot_I & \xrightarrow{\quad ccc \quad} & \bot_I^1 \\ & {\scriptstyle i_{B^A}} \searrow \quad \swarrow {\scriptstyle (i_B)^{\circ A}} & \\ & B^A & \end{array}$$

  where $\Delta_A = (id_A, id_A) : A \to A \times A$.

- *$p_{A,B}$ is a central isomorphism.*

*$s_{A,B,C}$ is called the **exponential strength**, whereas $p_{A,B}$ is the **quantification strength**.*

Remark: The naturality of $s_{A,B,C}$ in A and B follows from its definition, as well as the naturality of $p_{A,B}$ in A and B.

2.3 Interpretation of the calculus

Thanks to the notion of control category, we are able to give a categorical interpretation of $\lambda\mu 2$.

Interpretation of types:

If $I = U^n$, we note $\parr_n$ for $\parr_I$, $\bot_n$ for $\bot_I$ and Π_n for Π_I. Each type A such that $\vec{X} \Vdash A$ is interpreted as an object A^* of $F(U^n)$ as follows:

$$\bot^* = \bot_n \qquad \top^* = 1 \qquad X_i^* = \pi_n^i$$
$$(A \times B)^* = A^* \times B^* \qquad (A \parr B)^* = A^* \parr_n B^* \qquad (A \to B)^* = (B^*)^{A^*}$$
$$(\forall X_{n+1}.A)^* = \Pi_n(A^*)$$

Note that the interpretation of X_i is a morphism $\pi_n^i : U^n \to U$ in the base category $|\mathbf{H}|$: actually, we use here the fact that the composition of F with the forgetful functor $fff : \mathbf{CCC} \to \mathbf{Set}$ generates the functor $|\mathbf{H}|(-, U)$. Hence, it is equivalent to define the interpretation of a type as an object in $F(U^n)$ or as a morphism from U^n to U in $|\mathbf{H}|$.

Lemma 1 *Let A and B be two types such that $FTV(A), FTV(B) \in \{X_1, \ldots, X_n\}$. We note $- \mapsto (-)[U^n, B]$ for $F(id_{U^n} \times B^*)$. Then $(A[B/X_{n+1}])^* = (A^*)[U^n, B]$.*

Proof: We prove it by a structural induction on A: as $F(id_{U^n} \times B^*)$ is a strict morphism of pre-control categories, the only cases to check are $A = \forall X_j.A'$ and $A = X_i$. The first case is ensured by the naturality of Π_n, whereas the second one only requires a direct verification. □

Interpretation of terms:

A typing judgement of the form $\vec{X}; \Gamma \vdash t : A \mid \Delta$ will be interpreted as a morphism $\Gamma^* \to A^* \parr_n \Delta^*$ in the category $F(U^n)$.

In what follows, for the sake of simplicity we use A instead of A^* when dealing with an object of the category $F(U^n)$.

$$
\begin{aligned}
[\![\vec{X};\Gamma \vdash x_i : B_i \mid \Delta]\!] &= \Gamma \xrightarrow{\pi_i} B_i \xrightarrow{w} B_i \parr_n \Delta \\
[\![\vec{X};\Gamma \vdash \star : \top \mid \Delta]\!] &= \Gamma \xrightarrow{\diamond} 1 \xrightarrow{\cong} 1 \parr_n \Delta \\
[\![\vec{X};\Gamma \vdash (t,u) : A \times B \mid \Delta]\!] &= \Gamma \xrightarrow{([\![t]\!],[\![u]\!])} (A \parr_n \Delta) \times (B \parr_n \Delta) \xrightarrow{d} (A \times B) \parr_n \Delta \\
[\![\vec{X};\Gamma \vdash \pi_1(t) : A \mid \Delta]\!] &= \Gamma \xrightarrow{[\![t]\!]} (A \times B) \parr_n \Delta \xrightarrow{\pi_1 \parr \Delta} A \parr_n \Delta \\
[\![\vec{X};\Gamma \vdash \pi_2(t) : B \mid \Delta]\!] &= \Gamma \xrightarrow{[\![t]\!]} (A \times B) \parr_n \Delta \xrightarrow{\pi_2 \parr \Delta} B \parr_n \Delta \\
[\![\vec{X};\Gamma \vdash tu : B \mid \Delta]\!] &= \Gamma \xrightarrow{([\![t]\!],[\![u]\!])} (B^A \parr_n \Delta) \times (A \parr_n \Delta) \xrightarrow{d} (B^A \times A) \parr_n \Delta \xrightarrow{\epsilon \parr \Delta} B \parr_n \Delta \\
[\![\vec{X};\Gamma \vdash \lambda x^A.t : A \to B \mid \Delta]\!] &= \Gamma \xrightarrow{\Lambda([\![t]\!])} (B \parr_n \Delta)^A \xrightarrow{s^{-1}} B^A \parr_n \Delta \\
[\![\vec{X};\Gamma \vdash [\alpha_i]t : \bot \mid \Delta]\!] &= \Gamma \xrightarrow{[\![t]\!]} A_i \parr_n \Delta \xrightarrow{w_i \parr \Delta} \Delta \parr_n \Delta \xrightarrow{\nabla} \Delta \xrightarrow{\cong} \bot_n \parr_n \Delta \\
[\![\vec{X};\Gamma \vdash \mu\alpha^A.t : A \mid \Delta]\!] &= \Gamma \xrightarrow{[\![t]\!]} \bot_n \parr_n A \parr_n \Delta \xrightarrow{\cong} A \parr_n \Delta \\
[\![\vec{X};\Gamma \vdash [\alpha_i,\alpha_j]t : \bot \mid \Delta]\!] &= \Gamma \xrightarrow{[\![t]\!]} A_i \parr_n A_j \parr_n \Delta \xrightarrow{w_i \parr w_j \parr \Delta} \Delta \parr_n \Delta \parr_n \Delta \xrightarrow{\nabla \parr \Delta;\nabla} \Delta \xrightarrow{\cong} \bot_n \parr_n \Delta
\end{aligned}
$$

$$
\begin{aligned}
[\![\vec{X};\Gamma \vdash \mu(\alpha^A,\beta^B).t : A \parr B \mid \Delta]\!] &= \Gamma \xrightarrow{[\![t]\!]} \bot_n \parr_n A \parr_n B \parr_n \Delta \xrightarrow{\cong} (A \parr_n B) \parr_n \Delta \\
[\![\vec{X};\Gamma \vdash \Lambda X.t : \forall X.A \mid \Delta]\!] &= \Gamma \xrightarrow{\kappa([\![t]\!])} \Pi_n(A \parr_{n+1} (\Delta)^{U^{n+1}}_{U^n}) \xrightarrow{p^{-1}} \Pi_n(A) \parr_n \Delta \\
[\![\vec{X};\Gamma \vdash t\{B\} : A[B/X] \mid \Delta]\!] &= \Gamma \xrightarrow{\kappa^{-1}([\![t]\!];p)[U^n,B]} A[U^n,B] \parr_n \Delta
\end{aligned}
$$

Theorem 1 (soundness) *The interpretation of second-order $\lambda\mu$-terms in a control hyperdoctrine is sound: for any couple of terms t, u such that $\vec{X};\Gamma \vdash t = u : A \mid \Delta$, we have $[\![\vec{X};\Gamma \vdash t : A \mid \Delta]\!] = [\![\vec{X};\Gamma \vdash u : A \mid \Delta]\!]$. Thus, every control hyperdoctrine is a model of $\lambda\mu 2$.*

The main steps of the proof of this theorem can be found in the appendix.

3 The Game Model

Game models have originally been introduced by Hyland-Ong and Nickau [HO00, Nic94], and Abramsky-Jagadeesan-Malacaria [AJM00], giving rise to two different paradigms. In this section, we introduce a game model for $\lambda\mu 2$ by choosing an HON-style of games. This requires to introduce a notion of *arena*, on which we have to define the notion of *play*.

3.1 Polymorphic arenas

In this section we will describe polymorphic arenas, i.e. the arborescent structure by which we are going to interpret types. We need to be very

precise in defining this structure, because the control hyperdoctrine structure we wish to obtain forces us to have equalities like $(A \parr B)[C/X] = A[C/X] \parr B[C/X]$, which are non-trivial in a purely geometrical structure.

Hence, the name of nodes will carry an information about how the arena has been built: for example, the arena $A \to (B \times C)$ will be very similar to $(A \to B) \times (A \to C)$, except that the nodes will carry the information that the product has been made "before" the arrow. Actually, our arenas will be so near to formulas that we are allowed to use this correspondence to define substitution: instead of introducing it as an operation on forests, we define the operation $A \mapsto A[C/X]$ as the transformation of an arena A described by a formula F into the arena $A[C/X]$ described by the formula $F[C/X]$. This trivial definition is not the original goal of our presentation of arenas, but it is one of its advantages.

This arborescent structure of arenas hides a structure of hyperforests (i.e. a forest with additional structure), as introduced by Dominic Hughes [Hug97]. This structure is more convenient for dealing with plays on arenas, but we cannot introduce it from the beginning because of the precision we want for the objects of our model.

Polymorphic arenas are built with the constructors $\parr$, $\times$ and $\neg$; the constructor $\to$ is introduced at the end.

Construction of arenas:

We consider the set of type variables $X, Y, \dots$ to be in bijection with $\mathbb{N}\backslash\{0\}$, and we will further write this set $\mathcal{X} = \{X_j \mid j > 0\}$.

We define the set $\mathcal{N}$ of **nodes**, based on the following grammar:

$$c ::= \star \mid x_i \mid x^{(j,c)} \mid \forall(c) \mid (c,1) \mid (c,2) \mid \neg(c) \mid (c \parr c) \mid (c,c,1) \mid (c,c,2)$$

for $i, j \in \mathbb{N}$. The nodes x_i (resp. x^q) will be called **free variables** (resp. **bound variables**).

For a given node c, we note $V(c)$ the sets of nodes of the form x_i, $x^{(j,c')}$ or $\star$ occurring in c (we call them the **variables** of c). We define on the grammar the operation $c \mapsto c[c'/x]$ (where x can be either $\star$, x_i or $x^{j,c''}$), which simply consists in replacing each occurrence of x in c by c', and the operation $c \mapsto c[\star]$ which consists in replacing each occurrence of a variable of $V(c)$ by $\star$.

A forest A is a set of nodes E_A, together with an order relation $\leq_A$ on E_A such that for every c in E_A, the set $\{c' \mid c' \leq c\}$ is finite and totally ordered by $\leq_A$. We note $a <_A b$ if $a \leq_A b$ and $a \neq b$, and we say that c is a root of A if there

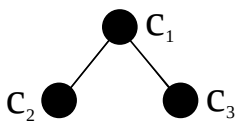
is no c' in E_A such that $c' <_A c$. Finally, for two nodes a and a' in a forest A such that $a \leq a'$, we note $d(a, a')$ the number of nodes x such that $a < x \leq a'$

If A and B are two forests, we define the following forests:

- $\top$, $\bot$ and X_i are defined by:
  - $E_\top = \emptyset \qquad E_\bot = \{\star\} \qquad E_{X_i} = x_i$
  - $<_\top$, $<_\bot$ and $<_{X_i}$ are the empty relations
- $A \times B$ is given by:
  - $E_{A\times B} = \{(a, 1) \mid a \in A\} \cup \{(b, 2) \mid b \in B\}$
  - $c <_{A\times B} c'$ iff $(c = (a, 1), c' = (a', 1)$ and $a <_A a')$ or $(c = (b, 1), c' = (b', 1)$ and $b <_B b')$
- $\neg A$ is given by:
  - $E_{\neg A} = \{\star\} \cup \{\neg(a) \mid a \in A\}$
  - $c <_{\neg A} c'$ iff $(c = \star$ and $c' \neq \star)$ or $(c = \neg(a), c' = \neg(a')$ and $a <_A a')$
- $A \parr B$ is given by:
  - $E_{A\parr B} = \{(a \parr b) \mid a$ root of $A \wedge b$ root of $B\} \cup \{(a, b_0, 1) \mid b_0$ root of $B \wedge \exists a_0 \in E_A, a_0 <_A a\} \cup \{(b, a_0, 2) \mid a_0$ root of $A \wedge \exists b_0 \in E_B, b_0 <_B b\}$
  - $c <_{A\parr B} c'$ iff $(c = (a \parr b), c' = (a', b, 1)$ and $a <_A a')$ or $(c = (a \parr b), c' = (b', a, 2)$ and $b <_B b')$ or $(c = (a, b, 1), c' = (a', b, 1)$ and $a <_A a')$ or $(c = (b, a, 2), c' = (b', a, 2)$ and $b <_B b')$
- $\forall X_i.A$ is given by:
  - $E_{\forall X_i.A} = \{\forall(a[x^{(0,a[\star])}/x_i]) \mid a$ root of $A\} \cup \{a[x^{(d(a_0,a),a_0[\star])}/x_i] \mid a_0$ root of $A \wedge a_0 <_A a\}$
  - $c <_{\forall X_i.A} c'$ iff $(c = a[x^{(d(a_0,a),a_0[\star])}/x_i], c' = a'[x^{(d(a_0,a'),a_0[\star])}/x_i]$ and $a_0 <_A a <_A a')$ or $(c = \forall(a[x^{(0,a[\star])}/x_i]), c' = a'[x^{(d(a,a'),a[\star])}/x_i]$ and $a < a')$

Remark: The variables $x^{(j,c)}$ correspond to bound type variables, and hence are related to an occurrence of $\forall$; but the challenge is to be able to characterize which one ! For this reason, they carry two pieces of information: first the distance (in the forest) of the node where this occurrence appears; second, the name of the node corresponding to this occurrence. Note that it is still normally not enough to say which occurrence of $\forall$ they are related to (think to $A \parr A$ for example), but thanks to the uniqueness of the construction of an arena, that we establish further, it becomes a sufficient information.

Example 1: Let us consider the arena $A = \forall X_3.(\neg\bot \parr X_3) \parr \forall X_3.(\neg X_2 \parr X_3)$. It can be represented graphically as follows:



with:

$c_1 = \forall(\star \parr x^{(0,\star\parr\star)}) \parr \forall(\star \parr x^{(0,\star\parr\star)})$

$c_2 = ((\neg(\star), x^{(0,\star\parr\star)}, 1), \forall(\star \parr x^{(0,\star\parr\star)})), 1)$

$c_3 = ((\neg(x_2), x^{(0,\star\parr\star)}, 1), \forall(\star \parr x^{(0,\star\parr\star)})), 2)$

There are many things to note: first, the occurrence of $x^{(0,\star\parr\star)}$ in c_2 and c_3 does not imply any dependency on the occurrences of $\forall$ in c_1. Moreover, one can remark that the two occurrences of $x^{(0,\star\parr\star)}$ in c_1 are a priori difficult to bind with a specific occurrence of $\forall$. However, there is no ambiguity if one can retrieve the way the arena has been built up.

All of this will be explicited by the following definitions. $\diamond$

The set $\mathcal{A}$ of **polymorphic arenas** is the smallest set of forests containing $\top$, $\bot$, X_i for $i \in \mathbb{N}$, and which is closed under the constructions product, lift, par and quantification. We note $FTV(A) = \{X_i \mid \exists c \in E_A, x_i \text{ appears in } c\}$. If $FTV(A) = \emptyset$, then A is called a **closed arena**. The set of closed arenas is denoted $\mathcal{H}$.

As a consequence of the definition, a polymorphic arena is described by a second-order formula built over $\neg$, $\parr$, $\times$. Actually, this description is essentially unique: let us define the congruence rule $=_\rho$ by:

- $\top \parr A =_\rho \top$
- $A \parr \top =_\rho \top$
- $\top \times \top =_\rho \top$
- $\neg\top =_\rho \bot$
- $\forall X_i.\top =_\rho \top$

The $\alpha\rho-$**equivalence** on formulas is the equivalence relation built over the congruence rule $=_\rho$ and the $\alpha-$equivalence.

Lemma 2 *Let A be a polymorphic arena, there exists a formula describing A. Furthermore, let F and F' be two such formulas, then F and F' are equal up to $\alpha\rho-$equivalence.*

Proof: This can be proved by induction on A. If A is empty, then either $F = F_1 \parr F_2$ with F_1 or F_2 describing an empty arena, or $F = \forall X.F'$ with F' describing an empty arena, or $F = F_1 \times F_2$ with F_1 and F_2 describing an empty arena (indeed, the other cases lead to non-empty arenas).

If A is not empty, let F be a formula describing A. Consider a root c of A:

- if $c = x_i$ then $F = X_i$
- if $c = \star$, then we have two possibilities: either $E_A = \{c\}$, and necessarily

$F = \bot$ or $F = \neg F'$ with F' describing an empty arena, or E_A is not reduced to c, and then $A = \neg A'$ for some non-empty $A' \in \mathcal{A}$, and we necessarily have $F = \neg F'$

- if $c = (c_1 \parr c_2)$, then we necessarily have $F = F_1 \parr F_2$, $A = A_1 \parr A_2$, and the names of nodes allows to recognize the arenas A_1 and A_2
- if $c = (c', 1)$ or $c = (c', 2)$, then we necessarily have $F = F_1 \times F_2$, $A = A_1 \times A_2$, and the names of nodes allows to recognize the arenas A_1 and A_2
- if $c = \forall(c')$ then each root c_n can be written $c_n = \forall(c'_n)$, and we necessarily have $F = \forall X_i.F(i)$, $A = \forall X_i.A(i)$ for some $X_i \notin FTV(A)$, where $A(i)$ is built by replacing each c_n by c'_n, and by replacing each occurrence of $x^{(j,c'_n[\star])}$ in a node $c_0 \geq_A c_n$ (with $d(c'_n, c_0) = j$) [1] by x_i. These occurrences $x^{(j,c'_n[\star])}$ are called the **bound variables** of the node c_n, and the nodes of A where they appear are called its **bound nodes**. By induction hypothesis, F is unique up to $\alpha\rho$-equivalence.

□

This result means that there is a one-to-one correspondence between arenas and $\alpha\rho$-equivalence classes of formulas.

Definition 10 (variable substitution) *Let A and B are two polymorphic arenas, respectively described by F_1 and F_2. We define the substitution $A[B/X_i]$ as the arena described by the formula $F_1[F_2/X_i]$.*

Note that this definition makes sense only because each arena corresponds to exactly one formula up to $\alpha\rho$-equivalence, and because the class of $\alpha\rho$-equivalence of $F_1[F_2/X_i]$ does not depend on the representatives we choose for F_1 and F_2.

Hyperforest:

Now that we have defined a notion of substitution on our arena, which trivially respects the required equalities like $(A \parr B)[C/X] = A[C/X] \parr B[C/X]$, $(A \times B)[C/X] = A[C/X] \times B[C/X]$, etc, we can express the structure of **hyperforests** that is hidden in these arenas: the notion presented here is directly inspired by Hughes' hyperforests which he introduced for his game model of system F. Actually, as we shall see further, this structure completely carries the equivalence relation corresponding to type isomorphisms.

For a given set E, $\mathbf{P}(E)$ is the set of *finite multisets* of elements of E. The multiset containing the elements $a_1, a_2, \ldots, a_n$ will be denoted $\{\{a_1, a_2, \ldots, a_n\}\}$.

[1] This is the reason why we specify the couple $q = (j, c)$ for a variable x^q: this allows us to determine exactly which are the variables x^q related to a specific occurrence of $\forall$.

Definition 11 (hyperforest) *An hyperforest $H = (E, \leq, \mathcal{R}, \mathcal{D})$ is a finite forest $(E, \leq)$, together with a multiset of **hyperedges** $\mathcal{R} \in \mathbf{P}(E \times \mathbf{P}(E))$ such that, for each $b = (S, t)$ in $\mathcal{R}$, we have $t \leq s$ whenever $s \in S$, and a function $\mathcal{D} : E \mapsto \mathbf{P}(X)$ which associates to each node its multiset of **decorations**.*

Definition 12 (quantifiers) *Let A be a polymorphic arena. For every $c \in E_A$, we define the multiset $var(c)$ by induction on c:*

- $var(x_i) = var(x^q) = var(\star) = \emptyset$
- $var((a \parr b)) = var(a) + var(b)$
- $var((a, a', 1)) = var((a, a', 2)) = var((a, 1)) = var((a, 2)) = var(\neg(a)) = var(a)$
- $var(\forall(a)) = var(a) + \{S\}$ *where S is the multiset* [2] *of bound nodes of $\forall(a)$.*

*The multiset $\mathcal{R}_A$ of **quantifiers** of an arena A is defined by $\mathcal{R}_A = \{\{(t, S) \mid t \in E_A \wedge S \in var(t)\}\}$. For any quantifier $b = (t, S)$, we note $\mathcal{T}(b) = t$ (the **target** of b) and $\mathcal{S}(b) = S$ (the **source** of b).*

Finally, for $c \in E_A$, we note $quant(c) = \{(c, S) \in \mathcal{R}_A\}$.

Definition 13 (free variable publisher) *Let A be a polymorphic arena. For every $c \in E_A$ and $i \in \mathbb{N}$, we define the multiset $\mathcal{D}_A(c)$, called the **free variable publisher**, by induction on c:*

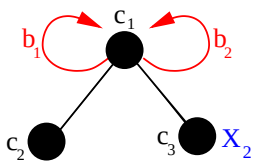
- $\mathcal{D}_A(x_i) = \{X_i\}$
- $\mathcal{D}_A(x^q) = \mathcal{D}_A(\star) = \emptyset$
- $\mathcal{D}_A((a \parr b)) = \mathcal{D}_A(a) + \mathcal{D}_A(b)$
- $\mathcal{D}_A((a, a', 1)) = \mathcal{D}_A((a, a', 2)) = \mathcal{D}_A((a, 1)) = \mathcal{D}_A((a, 2)) = \mathcal{D}_A(\forall(a)) = \mathcal{D}_A(\neg(a)) = \mathcal{D}_A(a)$

Lemma 3 *For any polymorphic arena A, $(E_A, \leq_A, \mathcal{R}_A, \mathcal{D}_A)$ is an hyperforest.*

Example 2: For the arena A defined in the preceding example, $var(c_1) = \{\{S_1, S_2\}\}$ where S_i contains only the ith occurrence of $x^{(0, \star \parr \star)}$. So, $\mathcal{R}_A = \{\{b_1, b_2\}\}$ with $b_1 = (c_1, \{\{c_1\}\})$ and $b_2 = (c_1, \{\{c_1\}\})$.

Besides, $\mathcal{D}_A(c_1) = \mathcal{D}_A(c_2) = \emptyset$ and $\mathcal{D}_A(c_3) = \{\{X_2\}\}$.

Hence, the hyperforest associated to the arena A can be represented graphically by:



[2] A node appears n times in S if it contains n bound variables of $\forall(a)$.

where straight lines stand for the relation $\leq_A$ (the smallest element is at the top), arrows stand for the hyperedges, and decorations are attached to nodes. $\diamond$

Substitution for a quantifier:

In order to define moves in an arena, we wish to give a definition of substitution for a quantifier, i.e. to define $A[B/b]$ for $A, B \in \mathcal{A}$ and $b \in \mathcal{R}_A$. According to the definition 12, the quantifier b is necessarily related to a specific occurrence of $\forall$ in the node $\mathcal{T}(b)$. Moreover, we know from the proof of lemma 2 that this occurrence of $\forall$ is itself related to a subformula $C = \forall X_i.C(i)$ in the formula describing A. Then we define the arena A', which is described by the formula A where C has been substituted by $C(i)$ (with $X_i \notin FTV(A)$), and we set $A[B/b] = A'[B/X_i]$ (note that this definition does not depend on the choice of i).

Origin:

As our arenas are nearly equivalent to formulas, we introduced the substitution through formulas. However, we could have given an explicit formulation of the arena $A[B/X_i]$, starting from A and B. But this formulation would have been very technical, whereas for our model we just need to know that each node of $A[B/X_i]$ is related to a particular node of A. This idea is expressed in the following lemma. The notation $[\alpha(x)/x]_{x\in V(a)}$ with $\alpha : V(a) \to \mathcal{N}$ indicates successive substitutions $[\alpha(x)/x]$ for x varying in $V(a)$.

Lemma 4 *Let A, B be two polymorphic arenas, and c a node of $E_{A[B/X_i]}$. Then there exists a unique node $a \in E_A$ and a function $\alpha : V(a) \to \mathcal{N}$ such that $c = a[\alpha(x)/x]_{x\in V(a)}$.*

*The node a is called the **origin** of c in A, and denoted $origin(c)$.*

Proof: First we prove the uniqueness of the node a: suppose $a, b \in E_A$ with $a[\alpha(x)/x]_{x\in V(a)} = b[\beta(x)/x]_{x\in V(a)}$. Then an induction on A ensures that $a = b$:

- if $A = X_i$ or $A = \bot$ it is obvious
- if $A = \forall X_i.A_0$ we have $a = \forall(a_0)$ and $b = \forall(b_0)$ with $a_0[\alpha(x)/x]_{x\in V(a_0)} = b_0[\beta(x)/x]_{x\in V(b_0)}$
- if $A = A_1 \parr A_2$ then $a = a_1 \parr a_2$ or $a = (a_1, a_2, 1)$ or $a = (a_1, a_2, 2)$. Then we have respectively $b = b_1 \parr b_2$ or $a = (b_1, b_2, 1)$ or $a = (b_1, b_2, 2)$, with $a_1[\alpha(x)/x]_{x\in V(a_1)} = b_1[\beta(x)/x]_{x\in V(b_1)}$ and $a_2[\alpha(x)/x]_{x\in V(a_2)} = b_2[\beta(x)/x]_{x\in V(b_2)}$
- the other cases are similar.

The existence of a can be proved by a structural induction on c:

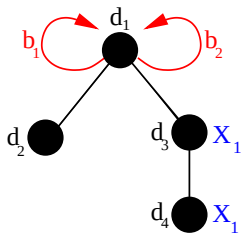
- if $c = x_i$ or $c = \star$ or $c = x^{(j,q)}$ the proof is trivial
- if $c = (c', 1)$ or $c = (c', 2)$ then $A[B/X_i] = G_1 \times G_2$, and from the definition of substitution we deduce that either $A = X_i$ or $A = A_1 \times A_2$ with $G_1 = A_1[B/X_i]$ and $G_2 = A_2[B/X_i]$; in this case c' is a node of G_1 or G_2, hence $c' = a_0[\alpha'(x)/x]_{x \in V(a_0)}$ with a_0 node of A_1 or A_2
- if $c = \neg(c')$ then $A[B/X_i] = \neg G$, and from the definition of substitution we deduce that either $A = X_i$ or $A = \neg A'$ with $G = A'[B/X_i]$; in this case c' is a node of G, hence $c' = a_0[\alpha'(x)/x]_{x \in V(a_0)}$ with a_0 node of A'
- if $c = c_1 \parr c_2$ then $A[B/X_i] = G_1 \parr G_2$, and from the definition of substitution we deduce that either $A = X_i$ or $A = A_1 \parr A_2$ with $G_1 = A_1[B/X_i]$ and $G_2 = A_2[B/X_i]$; in this case c_1 is a root of G_1 and c_2 is a root of G_2, hence $c_1 = a_1[\alpha_1(x)/x]_{x \in V(a_1)}$ with a_1 root of A_1, and $c_2 = a_2[\alpha_2(x)/x]_{x \in V(a_2)}$ with a_2 root of A_2
- if $c = (c_1, c_2, 1)$ then $A[B/X_i] = G_1 \parr G_2$, and from the definition of substitution we deduce that either $A = X_i$ or $A = A_1 \parr A_2$ with $G_1 = A_1[B/X_i]$ and $G_2 = A_2[B/X_i]$; in this case c_1 is a node of G_1 and c_2 is a root of G_2, hence by induction hypothesis $c_1 = a_1[\alpha_1(x)/x]_{x \in V(a_1)}$ with a_1 node of A_1, and $c_2 = a_2[\alpha_2(x)/x]_{x \in V(a_2)}$ with a_2 root of A_2
- if $c = (c_1, c_2, 2)$ then $A[B/X_i] = G_1 \parr G_2$, and from the definition of substitution we deduce that either $A = X_i$ or $A = A_1 \parr A_2$ with $G_1 = A_1[B/X_i]$ and $G_2 = A_2[B/X_i]$; in this case c_1 is a node of G_2, c_2 is a root of G_1 and $n_{c_2} < n_c$, hence by induction hypothesis $c_1 = a_1[\alpha_1(x)/x]_{x \in V(a_1)}$ with a_1 node of A_2, and $c_2 = a_2[\alpha_2(x)/x]_{x \in V(a_2)}$ with a_2 root of A_1
- if $c = \forall(c')$ then $A[B/X_i] = \forall X_k.G$ for some $k \in \mathbb{N}$, and from the definition of substitution we deduce that either $A = X_i$ or $A = \forall X_{k'}.A'$. In this case, thanks to α-equivalence, one can choose $k = k' \neq i$, and so $G = A[B/X_i]$; c' is a node of G, hence $c' = a_0[\alpha'(x)/x]_{x \in V(a_0)}$ with a_0 node of A'.

□

As a consequence, one can also establish a notion of origin for the substitution $A \mapsto A[B/b]$, defined similarly: for any node c of $E_{A[B/X_i]}$, there exists a unique node $a \in E_A$ and a function $\alpha : V(a) \to \mathcal{N}$ such that $c = a[\alpha(x)/x]_{x \in V(a)}$, or $c = a'[\alpha(x)/x]_{x \in V(a)}$ where a' is obtained from a by erasing one occurrence of $\forall$.

Definition 14 (offspring) *Let A, B be two polymorphic arenas, c node of $E_{A[B/X_i]}$ (resp. of $E_{A[B/b]}$) and $a = origin(c)$. We say that c is an **offspring** of a in $A[B/X_i]$ (resp. $A[B/b]$) if c is minimal among the c' such that $a = origin(c')$.*

Example 3: Let us consider the arena $A = \forall X_3.(\neg\bot \parr X_3) \parr \forall X_3.(\neg X_2 \parr X_3)$ from the preceding examples, and $B = (\neg X_1) \parr X_1$. Then $A[B/X_2] = \forall X_3.(\neg\bot \parr X_3) \parr \forall X_3.(\neg((\neg X_1) \parr X_1) \parr X_3)$ can be represented as follows:



with:

$$d_1 = \forall(\star \parr x^{(0,\star\parr\star)}) \parr \forall(\star \parr x^{(0,\star\parr\star)})$$

$$d_2 = ((\neg(\star), x^{(0,\star\parr\star)}, 1), \forall(\star \parr x^{(0,\star\parr\star)})), 1)$$

$$d_3 = ((\neg(\star \parr x_1), x^{(0,\star\parr\star)}, 1), \forall(\star \parr x^{(0,\star\parr\star)})), 2)$$

$$d_4 = ((\neg(\neq x_1, x_1, 1), x^{(0,\star\parr\star)}, 1), \forall(\star \parr x^{(0,\star\parr\star)})), 2)$$

Consider for example the node d_3: we can write $d_3 = a[b/x_2]$ with $a = ((\neg(x_2), x^{(0,\star\parr\star)}, 1), \forall(\star \parr x^{(0,\star\parr\star)})), 2) \in E_A$ and $b = \star \parr x_1 \in E_B$. Hence, a is the origin of d_3, i.e. the part of the move played in A and b is the part played in B. Similarily, $d_4 = a[c/x_2]$ with $c = (\neq x_1, x_1, 1) \in E_B$, so the origin of d_4 is also a. But d_3 is an offspring of a whereas d_4 is not: indeed, d_3 is the minimal node in $A[B/X]$ whose origin is a. $\diamond$

Remark: For the rest of this article, we introduce the constructor $A \to B = (\neg A) \parr B$. We will generally identify B (resp. A) to its trivially isomorphic part (resp. to its copies) in $A \to B$.

3.2 The notion of game in a polymorphic arena

We now informally describe a **play** in a polymorphic arena A, with parameters $\vec{X} = (X_1, \ldots, X_n)$ (this parameters will correspond to the free type variables appearing in a term).

As in propositional HON-games, a play is a sequence of **moves**, played alternately by two players: **P** (the Player) and **O** (the Opponent). But this time, because of second-order, playing a move does not simply consist in choosing a node in the arena: it is a more complicated process. Each move follows different steps :

- one choose a node in the forest given by the moves justifying this move
- one **instantiates** all the quantifiers whose target are the chosen node
- one substitutes the arena where we are playing
- if necessary, one chooses a new node in the arena we obtained, and one iterates the process.

Let us take a look further at how the process is actually working.

A **move** m is played in a closed arena H by **P** or **O**, who begins by choosing a node c_1 in $B^0 = H$, and then **instantiates** by a closed arena each quantifier b such that $\mathcal{T}(b) = c_1$. This modifies the arena consequently: each node $d \in \mathcal{S}(b)$ from B^0 is replaced by the closed arena instantiating b, this gives us the closed arena B^1.

But what happens if $c_1 \in \mathcal{S}(b)$? Once the node has been replaced by its instantiation G, one may have a doubt on the node the player really chose: for example, if G contains two trees, one must say which of the two roots has been chosen. Moreover, if the chosen node contains quantifiers, they have to be instantiated as well. So, the player has to choose another node c_2 in B^1, and possibly to instantiate the corresponding quantifiers to obtain a new closed arena B^2, etc. This process is an "horizontal" enlarging [3], in the sense that one does not go deeper in the closed arena, but one makes it evolve until there is no ambiguity on the chosen node. The player finally stops on a node c_n such that there is no b for which $c_n \in \mathcal{S}(b)$. c_n is called the **resulting node** of m, and the closed arena B^n is its **resulting arena**.

As in a propositional setting, a play in a polymorphic arena A is a sequence of moves with a relation of **justification**, but this time if m_i justifies m_j we require m_j to be played in the resulting arena of m_i: consequently, a move in a play can be chosen only when the quantifiers above it has been instantiated. Moreover, each initial move begins with a function $\theta : \{X_1, \ldots, X_n\} \to \mathcal{H}$ which instantiates every variable X_i by a closed arena; the player replaces in the arena A each node decorated by X_i by the closed arena $\theta(X_i)$, and then plays a move beginning with a root in the closed arena finally obtained: this allows to play in any polymorphic arena instead of just in a closed one.

Explicit examples will come after the definition to make all these intuitions clearer.

3.3 Moves, plays and strategies

Definition 15 (move) *A **move** in a closed arena H takes the form*

$$m = [c_1 : A_1^1/b_1^1; A_2^1/b_2^1; \ldots; A_{k_1}^1/b_{k_1}^1][c_2 : A_1^2/b_1^2; \ldots; A_{k_2}^2/b_{k_2}^2] \ldots\ldots$$
$$[c_n : A_1^n/b_1^n; \ldots; A_{k_n}^n/b_{k_n}^n]$$

with the following conditions:

[3] In $\lambda\mu 2$, this horizontal enlarging would correspond to a term with successive type instantiations, like $t\{\forall X.X\}\{\forall Y.\bot \to Y\}\{\forall Z.Z \to Z\}$....

- *c_1 is a node of $B^1 = H$ (called the **first node** of m) such that $quant(c_1) = \{b^1_1, \ldots, b^1_{k_1}\}$; we note $B^2 = H[A^1_1/b^1_1, \ldots, A^1_{k_1}/b^1_{k_1}]$*
- *for all $r \in [2, n]$, c_r is a node of B^r such that c_r offspring of c_{r-1} in B^r and $quant(c_r) = \{b^r_1, \ldots, b^r_{k_r}\}$; we note $B^{r+1} = B^r[A^r_1/b^r_1, \ldots, A^r_{k_r}/b^r_{k_r}]$*
- *for all $r \in [1, n-1]$, there exists b_l such that $c_r \in \mathcal{S}(b_l)$ in B^r; for c_n, there is no b_l such that $c_n \in \mathcal{S}(b_l)$ in B^n*

*We call **resulting arena** of m the closed arena B^{n+1}, and **resulting move** the move c_n, which is a root of B^n.*

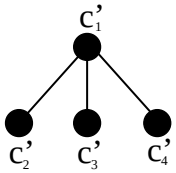
Remark: In order to avoid any confusion, take care to the fact that, in this model, the notions of *node* and *move* do not coincide as they did in the propositional model.

Definition 16 (initial move) *An **initial move** in a polymorphic arena A with parameters $\vec{X} = (X_1, \ldots, X_n)$ ($FTV(A) \subseteq \vec{X}$) takes the form $m = \theta m'$ where $\theta : \{X_1, \ldots, X_n\} \to \mathcal{H}$ and m' is a move in the closed arena $H = A[\theta(X_1)/X_1] \ldots [\theta(X_n)/X_n]$ such that the first node of m' is a root of H.*

Example 4: Consider the arena $A = \forall X_3.(\neg\bot \parr X_3) \parr \forall X_3.(\neg X_2 \parr X_3)$ described in the example 1, and the arenas $H_1 = \forall X.\neg X$, $H_2 = \bot$, $H_3 = \bot$.

Then $m = \theta[c_1 : H_1/b_1, H_2/b_2][c' : H_3/b_3]$, where $\theta(X_1) = \theta(X_2) = \bot$, c' is the root of $A' = ((\neg\bot) \parr (\forall X_4.\neg X_4)) \parr ((\neg\bot) \parr \bot)$ and b_3 is the unique quantifier of A', is an initial move in A with parameters $\vec{X} = (X_1, X_2)$.

The resulting arena of this move is $A'' = ((\neg\bot) \parr (\neg\bot)) \parr ((\neg\bot) \parr \bot)$, which can be represented graphically as:



with:

$c'_1 = (\star \parr \star) \parr (\star \parr \star)$

$c'_2 = ((\neg(\star), \star, 1), \star \parr \star, 1)$

$c'_3 = ((\neg(\star), \star, 2), \star \parr \star, 1)$

$c'_4 = ((\neg(\star), \star, 1), \star \parr \star, 2)$

$\diamond$

With these definitions, the internal structure of moves carries all the second-

order complexity, so that the external structure will now take the same form as in a propositional setting: the definitions of plays, strategies, etc, will be the standard ones.

Definition 17 (justified sequence, play) *A **justified sequence** in a polymorphic arena A with parameters $\vec{X}$ is a finite sequence of (initial and non-initial) moves $s = m_1 \dots m_n$, together with a partial function $ref : \{m_1, \dots, m_n\} \rightharpoonup \{m_1, \dots, m_n\}$ such that* [4] *:*

- *if $ref(m_i)$ is not defined, then m_i is an initial move in A with parameters $\vec{X}$*
- *if $ref(m_j) = m_i$ then $i < j$ and m_j non-initial move in the closed arena H^i (resulting arena of m_i) such that its first node is a son of the resulting node of m_i.*

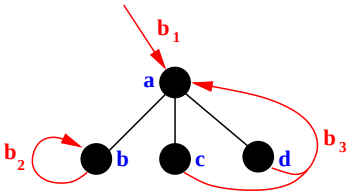
In a justified sequence s, we say that a move m is played by **P** *(resp. by* **O***) if the greatest natural number n such that $ref^n(m)$ is defined is odd (resp. even); then we note $\lambda(m) =$* **P** *(resp. $\lambda(m) =$* **O***).*

*A **play** is a justified sequence $s = m_1 \dots m_n$ such that, for all $1 \leq j \leq n-1$, we have $\lambda(m_{j+1}) \neq \lambda(m_j)$. The set of plays on A with parameters $\vec{X}$ is denoted $\mathcal{P}_{\vec{X}}(A)$. A **thread** is a play $s = m_1 \dots m_n$ such that m_1 is the only initial move in s. A P-view (resp. an O-view) is a play $s = m_1 \dots m_n$ such that $ref(m_j) = m_{j-1}$ for each j odd (resp. even). A **bi-view** is both a P-view and an O-view.*

Example 5:
Consider the term $t = \Lambda Z.\Lambda X.\lambda x^{\forall Y.Y}.\lambda u^X.\lambda v^X.(x\{\forall U.U\}\{X \to \bot\})u$ of type $A = \forall Z \forall X.(\forall Y.Y) \to X \to X \to \bot$. As we see further, this term will be interpreted in our model as a set of plays: we are going to describe one of these plays. For the sake of simplicity, we do not pay attention to the name of nodes, we only explicit the hyperforest structure associated to an arena.

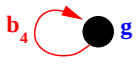
The starting arena is G_0, interpretation of A:

G_0 =

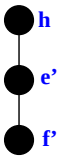


We define three other arenas:

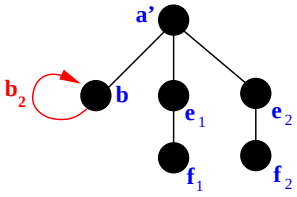
$H =$ 

$H' =$ 

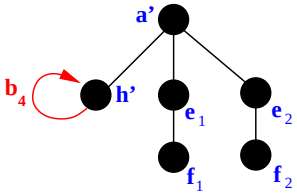
[4] Note that *ref* is actually a partial function from occurrences of moves to occurrences of moves.

$H'' =$ 

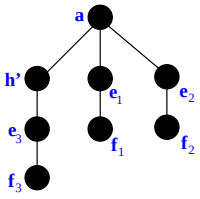
The first move of Opponent is $m_1 = \theta[a : H'/b_1; H/b_3]$ (θ does not play any role here) and it transforms G_0 into:

$G_1 =$ 

The second move is played by Player: it is written $m_2 = [b : H'/b_2][g' : H''/b_4][h' :]$ and it transforms G_1 successively in:

$G_2 =$ 

and in:

$G_3 =$ 

We now understand why it is important to have a sequence of nodes with their associated closed arenas: when we play $[b : H'/b_2]$, this corresponds to $x\{\forall U.U\}$, and when we play $[g' : H''/b_4]$ this corresponds to $\{X \to \bot\}$. Finally, $[h' :]$ simply indicates the last node we chose (there could be several choices if b_4 were instantiated by a product for example).

The rest of the play is a simple dialog between Opponent and Player in the arena G_3:

$$m_3 = [e_3 :]$$
$$m_4 = [e_1 :]$$
$$m_5 = [f_1 :]$$
$$m_6 = [f_3 :]$$

◇

Definition 18 (oldest ancestor) *The **oldest ancestor** of m in a play s, denoted $ref^{\infty}(m)$, is the move m' such that $ref^{n}(m) = m'$ for some n and $ref(m')$ is not defined: it is an initial move. We note θ_m the θ function appearing at the beginning of the move $ref^{\infty}(m)$.*

Let us consider the arena $A \to B = \neg A \parr B$, and let $s \in \mathcal{P}_{\vec{X}}(A \to B)$ and m be a move of s. Let d be the origin of the first node of m. There can be two cases: either $d \in B$, in which case each node appearing in m is written $(c_1, \star, 2)$; then we define the move $\tilde{m}$ by replacing each node $(c_1, \star, 2)$ by c_1. Or $d \in A$, in which case each node appearing in m is written $(\neg(c_1), c_2, 1)$; then we define the move $\tilde{m}$ by replacing each node $(\neg(c_1), c_2, 1)$ by c_1 and, if d is a root of A, by adding the function θ_m at the beginning of the move. These notations allow us to define the notion of restriction of a play:

Definition 19 (restriction) *Let $s \in \mathcal{P}_{\vec{X}}(A \to B)$. The **restriction** of s to A (resp. to B), denoted $s \upharpoonright_A$ (resp. $s \upharpoonright_B$), is the sequence of moves $\tilde{m_1}, \dots, \tilde{m_n}$ (with the same pointers as in s, wherever it is possible), where $m_1, \dots, m_n$ are the moves such that the origin of their first node is a node of A (resp. of B) : we say that these moves are **played** in A (resp. in B).*

With this definition, $s \upharpoonright_A$ (resp. $s \upharpoonright_B$) is a justified sequence in A (resp. in B) with parameters $\vec{X}$.

Definition 20 (strategy) *A **strategy** σ in an arena A with parameters $\vec{X}$, denoted $\sigma : A; \vec{X}$, is a non-empty set of even-length plays of $\mathcal{P}_{\vec{X}}(A)$, which is closed by even-length prefix and deterministic: if sa and sb are two plays of σ then $sa = sb$.*

Definition 21 (central strategy) *Let $\sigma : A \to B; \vec{X}$. σ is **central** if*

- *in each play of σ, for each initial move m there is exactly one move m' played in A and justified by m*
- *for each initial move m, there is a play $mm' \in \sigma$ with m' played in A.*

3.4 Cartesian closed structure

We now have the ingredients for our model: polymorphic arenas and strategies. Let us give some basic categorical structure on these objects.

Definition 22 (identity) *The **identity strategy** on A, $id_A : A \to A; \vec{X}$, is defined by $id_{A;\vec{X}} = \{s \in \mathcal{P}_{\vec{X}}(A_1 \to A_2) \mid \forall t \text{ even prefix of } s,\ t \upharpoonright_{A_1} = t \upharpoonright_{A_2}\}$ (where A_1 and A_2 stand for the two occurrences of A in $A \to A$).*

We let the reader check that this indeed defines a (central) strategy.

Definition 23 (composition) *Let A, B, C be three polymorphic arenas. An **interaction** on A, B, C with parameters $\vec{X}$ is a justified sequence on $(A \to B) \to C$ with parameters $\vec{X}$ such that $u \upharpoonright_{A,B} \in \mathcal{P}_{\vec{X}}(A \to B)$, $u \upharpoonright_{B,C} \in \mathcal{P}_{\vec{X}}(B \to C)$ and $u \upharpoonright_{A,C} \in \mathcal{P}_{\vec{X}}(A \to C)$. We note $int_{\vec{X}}(A, B, C)$ the set of such interactions. Let $\sigma : A \to B; \vec{X}$ and $\tau : B \to C; \vec{X}$, we call **composition** of σ and τ the set of plays $\sigma; \tau = \{u \upharpoonright_{A,C} |\ u \in int_{\vec{X}}(A, B, C),\ u \upharpoonright_{A,B} \in \sigma$ and $u \upharpoonright_{B,C} \in \tau\}$*

We shall now recover many properties which have already been proved in game semantics for the propositional case (see for example [Har99]). As we said, the inner structure of moves is very different in second-order games, but not the structure of plays and strategies, so that all reasonings in the propositional case will still be valid in this case. In order to reuse them directly instead of rewriting them, we establish a translation of second-order objects (polymorphic arenas, moves, ...) into a propositional setting with infinite forests: basically, the idea is to associate, to each occurrence of a move in a play, its non-empty bi-view, and to consider it as a "propositional" move (the prefix order on bi-views will give us the order relation in the arena).

Pay attention to the fact that this translation is only a tool, not a necessary construction, and that in particular it has nothing to do with the interpretation of $\lambda\mu 2$ in a control hyperdoctrine (although we chose the same notation).

Definition 24 (translation) *We consider one set of parameters $\vec{X} = (X_1, \ldots, X_n)$. Let A be a polymorphic arena such that $FTV(A) \subset \vec{X}$, and $\mathcal{BV}_A$ be the set of non-empty bi-views on A. If we note $\leq$ the prefix order on bi-views, then $[\![A]\!] = (\mathcal{BV}_A, \leq)$ is a (generally infinite) forest, called the **translation** of the arena A.*

*Let $s \in \mathcal{P}_{\vec{X}}(A)$ with $s = m_1 \ldots m_n$. To each occurrence m_i in s one can associate the unique bi-view v_i it belongs to, and this bi-view is a move in $[\![A]\!]$. Then $[\![s]\!] = v_1 \ldots v_n$ (with the same pointers as s) is a play in $[\![A]\!]$, called the **translation** of s. The **translation** of a set σ of plays on A is $[\![\sigma]\!] = \{[\![s]\!] \mid s \in \sigma\}$.*

Lemma 5 *(i) $[\![\sigma]\!]$ is a strategy on $[\![A]\!]$ if, and only if, σ is a strategy on A.*
(ii) If $[\![\sigma]\!] = [\![\tau]\!]$ then $\sigma = \tau$.

Proof:

(i) From the definition of the translation we immediately deduce that $[\![\sigma]\!]$ is a non-empty set of even-length plays, closed by even-length prefix if, and only if, it is the case for σ.
Besides, suppose σ is deterministic and $[\![sa]\!], [\![sb]\!] \in [\![\sigma]\!]$. Then $sa, sb \in$

σ, so $sa = sb$ and $[\![a]\!] = [\![b]\!]$. Reciprocally, if $[\![\sigma]\!]$ is deterministic and $sa, sb \in \sigma$, then $[\![sa]\!], [\![sb]\!] \in [\![\sigma]\!]$, so $[\![sa]\!] = [\![sb]\!]$ and $sa = sb$

(ii) It suffices to remark that the translation $s \mapsto [\![s]\!]$ on plays is injective.

□

But we have to take caution to the fact that we do not have $[\![A \to B]\!] = [\![A]\!] \to [\![B]\!]$. However, there is an isomorphism between $[\![A \to B]\!]$ and a subforest of $[\![A]\!] \to [\![B]\!]$: to each move μ of $[\![A \to B]\!]$ (which is in fact a non-empty bi-view $\mu = m_1 \dots m_n$ played on $A \to B$), one associates the move μ' defined in the following way:

- if $origin(m_n) \in B$, $\mu' = \mu$
- if $origin(m_n) \in A$, $\mu' = (\theta_{m_1} m_2) m_3 \dots m_n$ (which is indeed a bi-view in A).

In what follows, we will call this operation an **adaptation**.

We can extend this definition to a play: if $s = \mu_1 \dots m u_n$ then $s' = \mu'_1 \dots m u'_n$, and to a set of plays: $\sigma' = \{s' \mid s \in \sigma\}$, and we have:

Lemma 6 $[\![\sigma; \tau]\!]' = [\![\sigma]\!]'; [\![\tau]\!]'$

Proof: Note first that the first composition takes place in a second-order setting, whereas the second one is the usual propositional composition. Remark also that the part of $[\![A]\!] \to [\![B]\!]$ which is isomorphic to $[\![A \to B]\!]$ is composed of the nodes of $[\![B]\!]$ and of the copies of nodes of $[\![A]\!]$ which begin with the same θ function as their ancestor in $[\![B]\!]$.

Suppose $\sigma : A \to B$ and $\tau : B \to C$. Then $\alpha_1 = [\![\sigma; \tau]\!]'$ is a set of plays on $[\![A]\!] \to [\![C]\!]$, as well as $\alpha_2 = [\![\sigma]\!]'; [\![\tau]\!]'$.

For each $s \in \alpha_1$, $s = t'$, where t is such that there exists u played on $(A \to B) \to C$ verifying $t = [\![u \upharpoonright A, C]\!]$, $u \upharpoonright A, B \in \sigma$ and $u \upharpoonright B, C \in \tau$. By translating and adapting the moves of u, one obtain a justified sequence v played on $([\![A]\!] \to [\![B]\!]) \to [\![C]\!]$, such that $v \upharpoonright [\![A]\!], [\![C]\!] = s$, $v \upharpoonright [\![A]\!], [\![B]\!] = ([\![u \upharpoonright A, B]\!])'$ and $v \upharpoonright [\![B]\!], [\![C]\!] = ([\![u \upharpoonright B, C]\!])'$. Hence $v \upharpoonright [\![A]\!], [\![B]\!] \in ([\![\sigma]\!])'$, $v \upharpoonright [\![B]\!], [\![C]\!] \in ([\![\sigma]\!])'$ and $s \in \alpha_2$.

Conversely, if $s \in \alpha_2$, there exists u played on $([\![A]\!] \to [\![B]\!]) \to [\![C]\!]$ such that $s = u \upharpoonright [\![A]\!], [\![C]\!]$, $u \upharpoonright [\![A]\!], [\![B]\!] \in ([\![\sigma]\!])'$ and $u \upharpoonright [\![B]\!], [\![C]\!] \in ([\![\tau]\!])'$. The two last conditions forces u to be in the part of $([\![A]\!] \to [\![B]\!]) \to [\![C]\!]$ which is isomorphic to $[\![(A \to B) \to C]\!]$, so that u can be adapted to obtain a sequence of $[\![(A \to B) \to C]\!]$. By taking the inverse translation of this adaptation of u, one obtain a justified sequence v played on $(A \to B) \to C$ such that $v \upharpoonright A, B \in \sigma$, $v \upharpoonright B, C \in \tau$ and $v \upharpoonright_{A,B} = t$ with $s = [\![t]\!]'$. Hence $s \in \alpha_2$. □

This result allows us to manipulate the translation of composition easily, and this yields to the following results:

Lemma 7 *(i) If $\sigma : A \to B; \vec{X}$ and $\tau : B \to C; \vec{X}$, then $\sigma;\tau$ is a strategy on $A \to C$ with parameters $\vec{X}$.*
(ii) If $\sigma : A \to B; \vec{X}$, $\sigma; id_B = id_A; \sigma = \sigma$.
(iii) If $\sigma : A \to B; \vec{X}$, $\tau : B \to C; \vec{X}$ and $\rho : C \to D$, we have $(\sigma;\tau);\rho = \sigma;(\tau;\rho)$.

Proof: As an example, we prove the last assertion:

$[\![(\sigma;\tau);\rho]\!]' = [\![\sigma;\tau]\!]';[\![\rho]\!]' = [\![\sigma]\!]';[\![\tau]\!]';[\![\rho]\!]' = [\![\sigma]\!]';[\![\tau;\rho]\!]'$. As it is trivial that $\alpha' = \beta'$ iff $\alpha = \beta$, we have $[\![\alpha]\!]' = [\![\beta]\!]'$ iff $\alpha = \beta$, so $(\sigma;\tau);\rho = \sigma;(\tau;\rho)$. □

Finally, for each sequence of variables $\vec{X} = X_1, \ldots, X_n$, we obtain a category of games : objects are polymorphic arenas whose variables are chosen between $X_1, \ldots, X_n$, and morphisms are strategies on these arenas with parameters $\vec{X}$. We note $\mathcal{G}_0(X_1, \ldots, X_n)$ this category.

To obtain a cartesian closed structure, we add innocence:

Definition 25 *Let s be a play on an arena A, we define the view of s (which is indeed a P-view), denoted $\ulcorner s \urcorner$, by:*

- $\ulcorner \epsilon \urcorner = \epsilon$
- $\ulcorner sm \urcorner = m$ *if m is an initial move*
- $\ulcorner sm \urcorner = \ulcorner s \urcorner m$ *if m is a* **P***-move*
- $\ulcorner smtn \urcorner = \ulcorner sm \urcorner n$ *if n is an* **O***-move justified by m.*

A strategy $\sigma : A$ is called innocent if, for every play sn of σ, the justifier of n is in $\ulcorner s \urcorner$, and if we have: if $smn \in \sigma$, $t \in \sigma$, tm play in A and $\ulcorner sm \urcorner = \ulcorner tm \urcorner$ then $tmn \in \sigma$.

Note that the game isomorphisms between isomorphic arenas we have built in proposition 3 are innocent.

The structure of plays is preserved by translation, so that we have : $[\![\sigma]\!]$ is innocent if and only if σ is innocent. So, id_A is innocent and if σ and τ are innocent then $\sigma;\tau$ is innocent.

Thanks to these properties, we obtain a subcategory of $\mathcal{G}_0(X_1, \ldots, X_n)$ by considering only innocent strategies. We denote this subcategory $\mathcal{G}(X_1, \ldots, X_n)$.

Proposition 1 *$\mathcal{G}(X_1, \ldots, X_n)$ is a cartesian closed category.*

Proof: Let A and B be two arenas in propositional game semantics, we note $A \cong B$ when there is an isomorphism between the forests A and B.

We have isomorphisms $[\![A \times B]\!] \cong [\![A]\!] \times [\![B]\!]$ and $[\![A \to B]\!] \cong [\![A]\!] \to [\![B]\!]$, that we treat as equalities.

We introduce the projections $\pi_1 = \{s \in \mathcal{P}_{\vec{X}}(A \times B \to A) \mid s \restriction_{A\to A} \in id_A\}$, $\pi_2 = \{s \in \mathcal{P}_{\vec{X}}(A \times B \to A) \mid s \restriction_{B\to B} \in id_B\}$ and the product of strategies $\sigma \times \tau = \{s \in \mathcal{P}_{\vec{X}}((A \times C) \to (B \times D)) \mid s_{A\to B} \in \sigma \wedge s_{C\to D} \in \tau\}$, and we check that $[\![\pi_1]\!]' = \pi_1 : [\![A]\!] \times [\![B]\!] \to [\![A]\!]$, $[\![\pi_2]\!]' = \pi_2 : [\![A]\!] \times [\![B]\!] \to [\![B]\!]$ and $[\![\sigma \times \tau]\!]' = [\![\sigma]\!]' \times [\![\tau]\!]' : ([\![A]\!] \times [\![C]\!]) \to ([\![B]\!] \times [\![C]\!])$. Similarly, we introduce $A \to \sigma = \{s \in \mathcal{P}_{\vec{X}}((A \to B) \to (A \to C)) \mid s \restriction_{A\to A} \in id_A \wedge s \restriction_{B\to C} \in \sigma\}$ and we can check that $[\![A \to \sigma]\!] = [\![A]\!] \to [\![\sigma]\!]$.

All the commutative diagrams and unicity properties we need to check are guaranteed by lemmas 5 and 6. □

3.5 Construction of an hyperdoctrine

In this section, we set up the operations that will give rise to an hyperdoctrine.

The first ingredient of this construction is a base category $\mathbb{B}$: here it has as objects natural numbers and as morphism $n \to m$ the m-tuples $\langle A_1, \ldots, A_m \rangle$, where $A_i \in \mathcal{G}(X_1, \ldots, X_n)$ for $1 \leq i \leq m$. The composition in this category is substitution: if $\vec{A} = \langle A_1, \ldots, A_m \rangle : n \to m$ and $\vec{B} = \langle B_1, \ldots, B_n \rangle : k \to n$ then $\vec{A} \circ \vec{B} = \langle A_1[\vec{B}/\vec{X}], \ldots, A_m[\vec{B}/\vec{X}] \rangle : k \to m$.

We then have to define a functor $\mathcal{G} : \mathbb{B}^{op} \to \mathbf{CCC}$ (where $\mathbf{CCC}$ is the category of cartesian closed categories with strict morphisms of ccc's). We choose $\mathcal{G}(k) = \mathcal{G}(X_1, \ldots, X_k)$, and for each $\vec{C} : n \to m$ we define $\mathcal{G}(\vec{C}) : \mathcal{G}(m) \to \mathcal{G}(n)$ (the specialization functor):

- for every $A \in \mathcal{G}(m)$, we set $\mathcal{G}(\vec{C})(A) = A[\vec{C}]$
- for every $\sigma : A \to B; X_1, \ldots, X_m$, we set $\mathcal{G}(\vec{C})(\sigma) = \sigma[\vec{C}]$ where $\sigma[\vec{C}] : A[\vec{C}]; X_1, \ldots, X_n$ is defined by: $s \in \sigma[\vec{C}]$ if and only if $\tilde{s} \in \sigma$, where $\tilde{s}$ is obtained by replacing each initial move $m_{in} = \theta m'_{in}$ in s by $m''_{in} = \theta' m'_{in}$ with $\theta'(X_k) = C_k[\theta(X_1)/X_1, \ldots, \theta(X_m)/X_m]$.

We let the reader ascertain that if σ is an innocent strategy then $\sigma[\vec{C}]$ is an innocent strategy.

We check that this actually gives us a functor $\mathcal{G} : \mathbb{B}^{op} \to \mathbf{CCC}$: indeed, $\mathcal{G}(\vec{C}) : \mathcal{G}(m) \to \mathcal{G}(n)$ is a strict morphism of ccc's (we know that $(A \times B)[\vec{C}] = A[\vec{C}] \times B[\vec{C}]$, $(A \to B)[\vec{C}] = A[\vec{C}] \to B[\vec{C}]$, we have to check that $(\sigma \times \tau)[\vec{C}] = \sigma[\vec{C}] \times \tau[\vec{C}]$, $(\sigma; \tau)[\vec{C}] = \sigma[\vec{C}]; \tau[\vec{C}]$, etc...). And the composition coincides with

substitution: $\mathcal{G}(\vec{C}) \circ \mathcal{G}(\vec{C'}) = \mathcal{G}(\vec{C'}[\vec{C}])$.

One can also check easily that composing this functor with the forgetful functor ob : **CCC** $\rightarrow$ **Set** gives us the representable functor $\mathbb{B}(-, 1)$.

For what follows, we need an additional definition:

Lemma 8 *Let m be a move in an arena H, such that $H = (A \rightarrow B)[C/X]$ for some arenas A, B and C. Suppose that the origin of m is a root of A (resp. of B). Then there is a unique move m' in the arena $(\forall X.A) \rightarrow B[C/X]$ (resp. $A[C/X] \rightarrow (\forall X.B)$) such that:*

- *the arenas occurring in m' are the same as the ones occurring in m, plus the arena C*
- *the resulting arena and the resulting nodes of m and m' are the same.*

*The move m' is called the **abstraction** of m along $A \rightarrow B$.*

Proof: Consider the case where m is played in B and suppose $m = \theta[c_1 : A_1^1/b_1^1; A_2^1/b_2^1; \ldots; A_{k_1}^1/b_{k_1}^1] \ldots [c_n : A_1^n/b_1^n; \ldots; A_{k_n}^n/b_{k_n}^n]$. We define $c_0 = origin(c_1)$ and b_0 the quantifier of target c_0 corresponding to $\forall X$ in $\forall X.B$.

If $X_{n+1} \notin \mathcal{D}(c_0)$, then $m' = \theta'[c_1 : C/b_0; A_1^1/b_1^1; \ldots; A_{k_1}^1/b_{k_1}^1] \ldots [c_n : A_1^n/b_1^n; \ldots; A_{k_n}^n/b_{k_n}^n]$. Otherwise, let c'_1 be the origin of c_1 in $B^1 = (A \rightarrow B)[\theta(X_k)/X_k]_k$ (i.e. *before* the substitution C/X). The quantifers among $b_1^1, \ldots, b_{k_1}^1$ which come from C are named $b_{i_1^1}^1, \ldots, b_{i_{p1}^1}^1$, the others are named $b_{j_1^1}^1, \ldots, b_{j_{q1}^1}^1$. Let c'_2 be the origin of c_2 in $B^2 = B^1[A_{i_1^1}^1/b_{i_1^1}^1, \ldots, A_{i_{p1}^1}^1/b_{i_{p1}^1}^1]$ (i.e.*before* the substitutions $A_{j_1^1}^1/b_{j_1^1}^1, \ldots, A_{j_{q1}^1}^1/b_{j_{q1}^1}^1$). The quantifers among $b_1^2, \ldots, b_{k_2}^2$ which come from $A_{i_1^1}^1, \ldots, A_{i_{p1}^1}^1$ are named $b_{i_1^2}^2, \ldots, b_{i_{p2}^2}^2$, the others are named $b_{j_1^2}^2, \ldots, b_{j_{q2}^2}^2$, etc. By iterating this process, we finally get to a node c'_n which is the origin of c_n before some substitutions $A_{j_1^{n-1}}^{n-1}/b_{j_1^{n-1}}^{n-1}, \ldots, A_{j_{q(n-1)}^{n-1}}^{n-1}/b_{j_{q(n-1)}^{n-1}}^{n-1}$. The quantifers among $b_1^n, \ldots, b_{k_n}^n$ which come from $A_{i_1^{n-1}}^{n-1}, \ldots, A_{i_{p(n-1)}^{n-1}}^{n-1}$ are named $b_{i_1^n}^n, \ldots, b_{i_{pn}^n}^n$, the others are named $b_{j_1^n}^n, \ldots, b_{j_{qn}^n}^n$. If $pn = 0$ then $m' = \theta[c'_1 : C/b_0; A_{j_1^1}^1/b_{j_1^1}^1; \ldots; A_{j_{q1}^1}^1/b_{j_{q1}^1}^1] \ldots [c'_n : A_{j_1^n}^n/b_{j_1^n}^n; \ldots; A_{j_{qn}^n}^1/b_{j_{qn}^n}^n]$; if $pn \neq 0$ then $m' = \theta[c'_1 : C/b_0; A_{j_1^1}^1/b_{j_1^1}^1; \ldots; A_{j_{q1}^1}^1/b_{j_{q1}^1}^1] \ldots [c_n : A_{i_1^n}^n/b_{i_1^n}^n; \ldots; A_{i_{pn}^n}^1/b_{i_{pn}^n}^n]$.

We let the reader check that the move defined above is in $(A[C/X] \rightarrow (\forall X.B)$. Note that this technical definition is just the natural way to define a node of $A[B/X] \rightarrow (\forall X.B)$, starting from a move in $A[B/X] \rightarrow B[C/X]$.

The case where m is played in A is similar. □

In the category $\mathbb{B}$, the projection is $\vec{X} = \langle X_1, \ldots, X_n \rangle : n+1 \to n$. It gives us a functor $\vec{X}^* = \mathcal{G}(\vec{X}) : \mathcal{G}(n) \to \mathcal{G}(n+1)$. We have to find a right adjoint for $\vec{X}$, and for this we introduce the notion of morphism quantification:

Definition 26 (morphism quantification) *Let $\sigma : A \to B; X_1, \ldots, X_n, X_{n+1}$. We define the strategy $\forall\sigma : (\forall X_{n+1}.A) \to (\forall X_{n+1}.B); X_1, \ldots, X_n$ as the set of plays $\forall s$ for $s \in \sigma$, where $\forall s$ is defined from s via the following operations:*

- *each initial move $m = \theta m_B$ is replaced by $m' = \theta' m'_B$, where $\theta'(X_k) = \theta(X_k)$ for $1 \leq k \leq n$, and m'_B is the abstraction of m_B along $(A \to B)[\theta(X_1)/X_1, \ldots, \theta(X_n)/X_n]$*
- *each move m_A whose origin is a root of A is replaced by its abstraction along $(A \to B)[\theta(X_1)/X_1, \ldots, \theta(X_n)/X_n]$.*

Once again, we let the reader verify that if σ is an innocent strategy, then $\forall\sigma$ is an innocent strategy.

We can now define the functor $\Pi_n : \mathcal{G}(n+1) \to \mathcal{G}(n)$ by $\Pi_n(A) = \forall X_{n+1}.A$ and $\Pi_n(\sigma) = \forall\sigma$.

Proposition 2 *Π_n is a right adjoint of $\vec{X}^*$.*

Proof: We first have to establish for each $C \in \mathcal{G}(n)$ a bijection $\kappa : \mathcal{G}(n+1)(\vec{X}^*(C), A) \to \mathcal{G}(n)(C, \forall X_{n+1}A)$. We notice that $\vec{X}^*(C) = C[\vec{X}] = C$.

If $\sigma : C \to A; X_1, \ldots, X_n, X_{n+1}$, $\kappa(\sigma) = \{\kappa(s) \mid s \in \sigma\}$, where $\kappa(s)$ is obtained from s by replacing each initial move of the form $m = \theta m_0$ by an initial move $m' = \theta' m'_0$ in $C \to \forall X_{n+1}A$ such that $\theta'(X_k) = \theta(X_k) =$ for $1 \leq k \leq n$, and m'_0 is the abstraction of m_0 along $C \to A[\theta(X_1)/X_1, \ldots, \theta(X_n)/X_n]$.

We finally just need to check the naturality of this bijection, namely that $\tau; \kappa(\sigma) = \kappa(\vec{X}^*(\tau); \sigma)$ and $\kappa(\tau; \sigma) = \kappa(\tau); \Pi_n(\sigma)$. This comes directly from the action of $\vec{X}^*$ and Π_n on strategies. □

Lemma 9 *Π_n is natural in n: $\mathcal{G}(\vec{C}) \circ \Pi_n = \Pi_m \circ \mathcal{G}(\vec{C}, X_{m+1})$.*

Proof: This is easy to check for objects: for $A \in \mathcal{G}(n+1)$, the formulas representing respectively $(\forall X_{n+1}.A)[\vec{C}]$ and $\forall X_{m+1}.A[\vec{C}, X_{m+1}]$ are α-equivalent, hence the arenas are equal.

On morphisms, this requires to check that $(\forall\sigma)[\vec{C}] = \forall(\sigma[\vec{C}, X_{m+1}])$, which is easy because the substitution $\sigma \mapsto \sigma[\vec{C}]$ does only modify the θ function of the initial moves. □

One can now conclude, using the results of [See87, Pit88]:

Theorem 2 *The structure $\mathcal{M}$ defined by the base category $\mathbb{B}$ and the functor $\mathcal{G} : \mathbb{B}^{op} \to$ **CCC** is an hyperdoctrine, and therefore a model of system F.*

The interpretation of a type A in this model is a polymorphic arena A^*, whereas the interpretation of a typing derivation ending with the judgement $\vec{X}; x_1 : A_1, \ldots, x_n : A_n \vdash t : A$ is a strategy $\sigma_t : A_1^* \times \ldots \times A_n^* \to A^*; \vec{X}$.

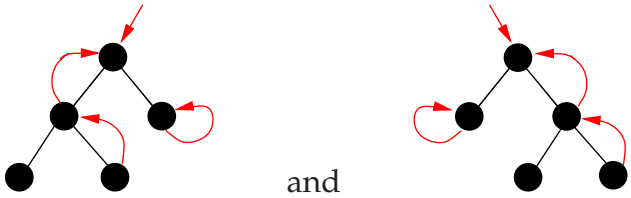
3.6 Arena isomorphisms

Before defining our control hyperdoctrine, we introduce the notions of arena isomorphisms and game isomorphisms that will be useful to define some structural morphisms. Note that these notions will become really important when dealing with type isomorphisms.

There are two ways to define an isomorphism between arenas: it can be either an isomorphism using strategies, or a (trivial) geometrical equality between hyperforests. We prove here that the first notion of isomorphism is implied by the latter.

Definition 27 (arena isomorphism) *Let A and B two polymorphic arenas. We say that there is an **arena isomorphism** between A and B if there is a bijection $g : E_A \to E_B$ preserving the hyperforest structure: $g(\mathcal{R}_A) = \mathcal{R}_B$ and $\mathcal{D}_B \circ g = \mathcal{D}_A$. We note this $g : A \simeq_a B$, or simply $A \simeq_a B$.*

Example 6: The arenas $A = \forall X.\forall Y.((\forall Z.(\bot \times Z) \to X) \times (\forall U.U)) \to \bot$ and $B = (\forall X.X) \to (\forall Y.(\forall Z.\bot \to Z \to Y) \to (\forall U.\bot))$ are isomorphic: indeed, they can be described by the following hyperforests:



and

◇

Definition 28 (game isomorphism) *Let A and B two polymorphic arenas. We say that there is a **game isomorphism** (σ, τ) between A and B ($A \simeq_g B$) if there are two strategies $\sigma : A \to B; \vec{X}$ and $\tau : B \to A; \vec{X}$ such that $\sigma; \tau = id_A$ and $\tau; \sigma = id_B$. We note this $(\sigma, \tau) : A \simeq_g B$, or simply $A \simeq_g B$.*

Proposition 3 *Let A and B two polymorphic arenas such that $A \simeq_a B$. Then there is a game isomorphism (σ, τ) between A and B; moreover, σ and τ are central*

strategies.

Proof: We wish to extend the function g into a function on plays.

First note that if $g : A \simeq_a B$ then $A[C/X_i] \simeq_a B[C/X_i]$ and, if $b \in \mathcal{R}_A$, $A[C/b] \simeq_a B[C/g(b)]$. We note $g[C/X_i]$ (or $g[C/b]$) the function realizing this isomorphism. For a given move $m = (\theta)[c_1 : A^1_1/b^1_1; \ldots; A^1_{k_1}/b^1_{k_1}] \ldots [c_p : A^p_1/b^p_1; \ldots; A^n_{k_p}/b^p_{k_p}]$ played in A (the notation (θ) indicates that a θ function may appear or not), we define $\tilde{g}(m) = (\theta)[g^1(c_1) : A^1_1/g^1(b^1_1); \ldots; A^1_{k_1}/g^1(b^1_{k_1})] \ldots [g^p(c_p) : A^p_1/g^p(b^p_1); \ldots; A^n_{k_p}/g^p(b^p_{k_p})]$ with $g^1 = g[\theta(X_1)/X_1] \ldots [\theta(X_n)/X_n]$ and $g^{i+1} = g^i[A^i_1/g^i(b^i_1)][A^i_{k_i}/g^i(b^i_{k_i})]$.

We note $\tilde{g}/m$ the function realizing the isomorphism between the resulting arenas of m and $\tilde{g}(m)$ (i.e. $\tilde{g} = g^{p+1}$). For a given play $s = m_1, \ldots m_n$, we define the functions g_j by: $g_1 = g$ and $g_{j+1} = \tilde{g_j}/m_j$. Finally, we set $\bar{g}(s) = \tilde{g_1}(m_1) \ldots \tilde{g_n}(m_n)$.

Consider $\sigma = \{s \in \mathcal{P}_{\vec{X}}(A \to B) \mid \forall t \textit{ even prefix of } s,\ t \upharpoonright_A = \bar{g}(t \upharpoonright_B)\}$ and $\tau = \{s \in \mathcal{P}_{\vec{X}}(B \to A) \mid \forall t \textit{ even prefix of } s,\ t \upharpoonright_B = \bar{g}(t \upharpoonright_A)\}$. Then σ and τ are indeed central strategies from on $A \to B$ and $B \to A$ respectively, and they verify $\sigma; \tau = id_A$ and $\tau; \sigma = id_B$.

The game isomorphisms we have constructed here will sometimes be called the **trivial isomorphisms** between A and B. □

The fundamental result of the third section of this article will be to prove that, in a certain submodel of this one, the converse of this proposition is also true.

3.7 Construction of a control hyperdoctrine

As we wish to establish that we have a model of the $\lambda\mu$2-calculus and we already have a structure of hyperdoctrine, we need to recover the additional requirements of the section 2.2. The important part of the job will in fact consist in characterizing $\parr$ as a binoidal functor, and distinguishing central morphisms.

Let s be a justified sequence on $A \parr B$; we want to define a justified sequence $s \upharpoonright_A$ on A. The basic idea is the following: each node c in $A \parr B$ "comes from" a node of A or B. $s \upharpoonright_A$ will consist of the part of s which comes from A.

Formally, if $s = \epsilon$ then $s \upharpoonright_A = \epsilon$; otherwise, let $s = s'm$ with $m = (\theta)[c_1 : A^1_1/b^1_1; \ldots; A^1_{k_1}/b^1_{k_1}] \ldots [c_n : A^n_1/b^n_1; \ldots; A^n_{k_n}/b^n_{k_n}]$. All the c_i are necessarily of the

same form: $c_i = (a_i \parr c'_i)$ or $c_i = (a_i, c'_i, 1)$ or $c_i = (c'_i, a_i, 2)$.

- If $c_i = (a_i \parr c'_i)$, let p be the least i such that $c'_{i+1} = c'_i$ ($i = n$ if this equality in never true), $b^1_{i_m}$ be the quantifiers appearing in $A[(\theta(X_j)/X_j)_j]$, and $b^l_{i_m}$ (for $2 \leq l \leq p$ and $1 \leq m \leq km$) be the quantifiers appearing in $A^{l-1}_{i_1}/b^{l-1}_{i_1}$, $\ldots$, $A^{l_1}_{i_{k(l-1)}}$; then $s \restriction_A = s' \restriction_A m'$ with $m' = \theta[a_1 : A^1_{i_1}/b^1_{i_1}; \ldots; A^1_{i_{k1}}/b^1_{ik1}] \ldots [a_n : A^p_{i_p}/b^p_{i_p}; \ldots; A^p_{i_{kp}}/b^n_{i_{kp}}]$
- If $c_i = (a_i, c'_i, 1)$, $s \restriction_A = s' \restriction_A m'$ with $m' = [a_1 : A^1_1/b^1_1; \ldots; A^1_{k_1}/b^1_{k_1}] \ldots [a_n : A^n_1/b^n_1; \ldots; A^n_{k_n}/b^n_{k_n}]$
- If $c_i = (c'_i, a_i, 2)$, $s \restriction_A = s' \restriction_A$.

$t \restriction_B$ is defined similarly.

Example 7: Let us come back to the arena A and the initial move m of example 4. We have $A = B \parr C$, with $B = \forall X_3.(\neg\bot \parr X_3)$ and $C = \forall X_3.(\neg X_2 \parr X_3)$, $m = \theta[c_1 : H_1/b_1, H_2/b_2][c' : H_3/b_3]$ with $c_1 = \forall(\star \parr x^{(0,\star\parr\star)}) \parr \forall(\star \parr x^{(0,\star\parr\star)})$ and $c' = c'_1 = (\star \parr \star) \parr (\star \parr \star)$.

If we consider the play $s = m$, one has $s \restriction_B = \theta[\forall(\star\parr x^{(0,\star\parr\star)}) : H_1/b_1][(\star\parr\star) : H_3/b_3]$ and $s \restriction_B = [\forall(\star\parr x^{(0,\star\parr\star)}) : H_2/b_2][(\star\parr\star) :]$. The reader can check that one has kept in $s \restriction_B$ the "left part" of the moves $a \parr a'$, and every instantiation of a quantifier which is related to the "left part" of the arena A. $\diamond$

This definition is such that, for s justified sequence on $A \parr B$, one has $s \restriction_A$ justified sequence on A. If s is a justified sequence on $(A \parr C) \to (B \parr D)$, one can define as well $s \restriction_{A\to B}$, composed of the moves of $(s \restriction_{A\parr C}) \restriction_A$ and the moves of $(s \restriction_{B\parr D}) \restriction_B$.

Proposition 4 *Let $\tau : C \to D; \vec{X}$ be a strategy, and $\sigma : A \to B; \vec{X}$ a central strategy. Let $\sigma \parr \tau = \{s \in \mathcal{P}_{\vec{X}}((A \parr C) \to (B \parr D)) \mid s \restriction_{A\to B} \in \sigma \wedge s \restriction_{C\to D} \in \tau\}$. Then $\sigma \parr \tau$ is a strategy. If σ and τ are innocent then $\sigma \parr \tau$ is innocent. Moreover, we have $(\sigma \parr id_C); (id_B \parr \tau) = (id_A \parr \tau); (\sigma \parr id_D)$.*

Proof: Once again we make use of the translation into propositional game semantics. We recall that, in propositional game semantics, the nodes of an arena $A \parr B$ are denoted (a_0, b_0) for roots, $(a, b_0, 1)$ for copies of nodes of A and $(b, a_0, 2)$ for copies of nodes of B.

First, one can verify that $[\![\sigma]\!]$ is a central strategy (in the propositional game semantics) iff σ is a central strategy.

The correspondence $[\![A \parr B]\!] \cong [\![A]\!] \parr [\![B]\!]$ is true, but not completely trivial: actually, this is precisely given by the preceding definition: to each non-empty bi-view sm of $A \parr B$ (which is a node of $[\![A \parr B]\!]$) ending with the

move m, one associates a node μ in $[\![A]\!] \parr [\![B]\!]$ by proceeding by cases:

- if the nodes of m take the form $(a_i \parr b_i)$, then $\mu = (sm \upharpoonright_A, sm \upharpoonright_B)$
- if the nodes of m take the form $(a_i, b_i, 1)$, then $\mu = (sm \upharpoonright_A, b_i, 1)$
- if the nodes of m take the form $(b_i, a_i, 2)$, then $\mu = (sm \upharpoonright_B, a_i, 2)$.

This defines a bijection from the nodes of $[\![A \parr B]\!]$ to the nodes of $[\![A]\!] \parr [\![B]\!]$.

By extending this correspondence to plays and sets of plays, one obtain, for each set σ of plays on $A \parr B$, a set $[\![\sigma]\!]''$ on $[\![A]\!] \parr [\![B]\!]$, and we have: $[\![\sigma \upharpoonright_A]\!] = [\![\sigma]\!]'' \upharpoonright_{[\![A]\!]}$. For a strategy $\tau : C \to D; \vec{X}$ and a central strategy $\sigma : A \to B; \vec{X}$, one has

$$\begin{aligned}
[\![\sigma \parr \tau]\!]'' &= \{[\![s]\!]'' \mid s \in \mathcal{P}_{\vec{X}}((A \parr C) \to (B \parr D)) \wedge s \upharpoonright_{A \to B} \in \sigma \wedge s \upharpoonright_{C \to D} \in \tau\} \\
&= \{s \in \mathcal{P}_{\vec{X}}(([\![A]\!] \parr [\![C]\!]) \to ([\![B]\!] \parr [\![D]\!])) \mid s \upharpoonright_{[\![A]\!] \to [\![B]\!]} \in [\![\sigma]\!] \\
&\qquad \wedge s \upharpoonright_{[\![C]\!] \to [\![D]\!]} \in [\![\tau]\!]\} \\
&= [\![\sigma]\!]; [\![\tau]\!]
\end{aligned}$$

Now we can refer to what has been done for propositional game semantics [Lau02] and retrieve the expected results (thanks to lemmas 5 and 6): $[\![\sigma; \tau]\!]''$ is a strategy, so $\sigma; \tau$ is a strategy. If and τ are innocent, then $[\![\sigma; \tau]\!]''$ is innocent, so $\sigma; \tau$ is innocent. Finally, $[\![(\sigma \parr id_C); (id_B \parr \tau)]\!]'' = [\![(A \parr \tau); (\sigma \parr D)]\!]$, so $(\sigma \parr id_C); (id_B \parr \tau) = (A \parr \tau); (\sigma \parr D)$. □

Of course, $\tau \parr \sigma$ for any τ and a central σ is defined similarly. This gives us the following result:

Proposition 5 *In every category $\mathcal{G}(n)$, $\parr$ is a binoidal functor. Central morphisms for this structure are exactly central strategies.*

Proof: We define $\sigma \parr C = \sigma \parr id_C$ and $C \parr \sigma = id_C \parr \sigma$. As $[\![(\sigma; \tau) \parr C]\!] = [\![(\sigma \parr C); (\tau \parr C)]\!]$ and $[\![C \parr (\sigma; \tau)]\!] = [\![(C \parr \sigma); (C \parr \tau)]\!]$, $\parr$ is a binoidal functor.

Moreover,

$$\begin{aligned}
\sigma \text{ is central for the binoidal structure} \quad &\Leftrightarrow \quad [\![\sigma]\!]'' \text{ is central for the} \\
&\qquad\quad \text{(propositional) binoidal structure} \\
&\Leftrightarrow \quad [\![\sigma]\!]'' \text{ is a central strategy} \\
&\Leftrightarrow \quad \sigma \text{ is a central strategy}
\end{aligned}$$

□

Theorem 3 *$\mathcal{M}$ is a control hyperdoctrine.*

Proof: For each $n \in \mathbb{N}$, we define $\parr_n = \parr$ and $\bot_n = \bot$.

To construct the central isomorphisms $a_{A,B,C}$, l_A, r_A, $c_{A,B}$, we use the fact that there are trivial arena isomorphisms between the arenas they bind together: for example, there is an arena isomorphism between $(A \parr B) \parr C$ and $A \parr (B \parr C)$, and $a_{A,B,C}$ is defined to be the corresponding game isomorphism. The reader can check the naturality of these isomorphisms and the commutation of associated diagrams.

The symmetric monoid of an object A is defined in the following way:

- $i_A = \{\epsilon\} \cup \{mm' \mid m \text{ initial move of } A \wedge m' = (\neg(\star), c_n, 1) \wedge c_n \text{ last node of } m\}$
- $\nabla_A = \{s \in \mathcal{P}_{\vec{X}}(A_1 \parr A_2 \to A_0) \mid \forall t \leq s \text{ with } |t| \text{ even }, t \upharpoonright_{A_1 \to A_0} \in id_A \wedge t \upharpoonright_{A_2 \to A_0} \in id_A\}$

The reader can check the innocence of these strategies and the commutativity of the required diagrams.

At this point, we have proved that $\mathcal{M}$ is a symmetric premonoidal hyperdoctrine with codiagonals. $\mathcal{M}$ is also distributive: π_1 and π_2 are focal, and $(\pi \parr C, \pi_2 \parr C)$ is the trivial isomorphism between $(A \times B) \parr C$ and $(A \parr C) \times (B \parr C)$, and the empty strategy is the trivial isomorphism between $\top \parr C = \top$ and $\top$. Checking naturality is left to the reader.

Hypercentrality is easy to check: because of their definitions, κ and κ^{-1} preserve centrality. Besides, thanks to the way we have defined the substitution for an arena (through substitution of the formula) and for a strategy (through a simple operation on the θ function), the specialization functors commute with $\parr$; and the operation of substitution for strategies of course transforms trivial isomorphisms into trivial isomorphisms.

Finally, $s_{A,B,C}$ is the trivial isomorphism between $(A \to B) \parr C$ and $A \to (B \parr C)$ (because $\hat{\epsilon}_{A,B,C} : (B^A \parr C) \times A \xrightarrow{(B^A \parr C) \times (l; A \parr i)} (B^A \parr C) \times (A \parr C) \xrightarrow{\epsilon \parr C} B \parr C$ is trivially equivalent to $ev \parr C$), and $p_{A,B}$ is the trivial isomorphism between $(\forall X_n.(A \parr B)$ and $\forall X_n(A) \parr B$ (if $X_n \notin FTV(B)$). One again, the naturality of $s_{A,B,C}$ in C and the commutativity of additional diagrams are left to the reader. □

4 Characterization of isomorphisms

Having defined our model, we would like to use it to characterize second-order type isomorphisms. Unfortunately, there are too many isomorphisms in our model: for example, there exists an isomorphism (σ,τ) between $\forall X.\bot$ and $\forall X \forall Y.\bot$.

Indeed, the set of closed arenas $\mathcal{H}$ is countable, hence there exists a bijection $k : \mathcal{H} \to \mathcal{H} \times \mathcal{H}$ ($k = (k_1, k_2)$). Then the innocent strategy $\sigma : (\forall X.\bot) \to (\forall X \forall Y.\bot)$ can be defined by its views, which take the form $s_{H,H'} = [\forall(\forall(\star)) : H/b_1, H'/b_2][(\neg(\forall(\star)), \forall(\forall(\star)), 1) : k^{-1}(H, H')/b_0]$ for $H, H' \in \mathcal{H}$. Similarly, the innocent strategy $\tau : (\forall X \forall Y.\bot) \to (\forall X.\bot)$ can be defined by its views, which take the form $t_H = [\forall(\star) : H/b_0][(\neg(\forall(\forall(\star))), \forall(\star)), 1) : k_1(H)/b_1, k_2(H)/b_2]$ for $H \in \mathcal{H}$.

But the problem is, that this isomorphism does not exist in our language $\lambda\mu 2$ [5] !

In order to characterize type isomorphisms more precisely, we will introduce a new property, called **uniformity**, which tends to move the model nearer to the behavior of $\lambda\mu 2$ itself. In particular, this property will break down the high level of symmetry between **P** and **O** (as it is the case for innocence).

4.1 Uniformity

Definition 29 (rank) *Let $s \in \mathcal{P}_{\vec{X}}(A)$ and m a move of s which takes the form*

$$m = (\theta)[c_1 : A_1^1/b_1^1; \ldots; A_{k_1}^1/b_{k_1}^1], \ldots\ldots, [c_n : A_1^n/b_1^n; \ldots; A_{k_n}^n/b_{k_n}^n]$$

For each occurrence H of a closed arena in m ($H = A_i^j$ for some (i, j) or $H = \theta(X_i)$ for some X_i), we define the rank of H, denoted $rank_m(H)$, by [6] *:*

- *if $H = \theta(X_i)$ for some $X_i \in \vec{X}$, or $H = A_i^j$ with b_i^j quantifier of the arena where m is played,then $rank_m(H) = 1$*
- *if $H = A_i^j$ where b_i^j is an quantifier of a closed arena H' occurring in m before H, then $rank_m(H) = rank_m(H') + 1$.*

Definition 30 (paths and instantiation traces) *Let $s \in \mathcal{P}_{\vec{X}}(A)$. For every move m of s, we define:*

- *the **path** of m: $\underline{m} = c_1 \ldots c_n$*
- *the **instantiation traces** of m: $\overline{m}^i = B_1 \ldots B_p$ is the sequence of occurrences of arenas B_j appearing in m such that $rank_m(B_i) \leq i$.*

We note C *the set of paths and $\mathcal{I}$ the set of instantiation traces. If $s = m_1 \ldots m_r$, we note $\underline{s} = \underline{m_1} \ldots \underline{m_r}$ (ref is implicit in $\underline{s}$) and $\overline{s}^j = \overline{m_1}^j \ldots \overline{m_r}^j$ for $j \in \mathbb{N}$.*

[5] It exists in Curry-style system F, but our language is defined in the Church-style.

[6] Note that if m is not an initial move, then $rank_m(A_i^j) = j$.

The path of m is the description of the move without looking at the instantiated arenas. Instantiations traces are the sequences of arenas effectively instantiated. The existence of many instantiation traces depending on the node is required by the proof of theorem 5.

Beside the set $\mathcal{X}$ of variable names that can be used as parameters, we put another set $\mathcal{Y} = \{C_i \mid i \in \mathbb{N}\}$ where the C_i's will represent **holes**, whose destiny is to be replaced by a closed arena.

Definition 31 (arenas with holes) *An **arena with holes** is a polymorphic arena built on the set of free variables $\mathcal{X} \cup \mathcal{Y}$. The set of arenas with holes will be denoted $\mathcal{K}$.*

Remark: This definition means that, to define arenas with holes, we extend the grammar of nodes with the variables c_i for $i \in \mathbb{N}$.

Definition 32 (uniform strategy) *A strategy $\sigma : A; \vec{X}$ is called **uniform** if there exists a partial function $f : \mathsf{C}^* \rightharpoonup \mathsf{C}^*$, and a sequence of functions $F_1, \ldots, F_n, \cdots : \mathsf{C}^* \to \mathcal{K}^*$ such that, if $s \in \sigma$ and sm play in A, then: $smm' \in \sigma$ if and only if $\underline{smm'} = f(\underline{sm})$ and $\overline{m'}^i = F_i(\underline{sm})[\overline{sm}^i]$ for every $i \in \mathbb{N}$.*

The notation $F_i(\underline{sm})[\overline{sm}^i]$ means that the sequence of closed arenas $\overline{m'}^i$ is obtained first by building the sequence $F_i(\underline{sm})$ of arenas with holes, then by applying the substitution $[\overline{sm}^i(1)/C_1, \ldots, \overline{sm}^i(p)/C_p]$ in these arenas, where $\overline{sm}^i(k)$ stands for the kth arena occurring in $\overline{sm}^i$ (and p is the length of this sequence).

This way, the arenas with holes cannot depend on already instantiated arenas: these ones can only fill the holes to generate the arenas for the following moves. This corresponds to the fact that, in a term of $\lambda\mu 2$, **P** does not have a direct access to the instantiation of types by **O**, he can only reuse them (think about the term $\Lambda X.\lambda x^{\forall Y.Y}.x\{X \to \bot\}$ for example). Likewise, via the function f we see that the paths of the moves of **P** in a uniform strategy do not depend on already instantiated arenas, but only on the names of already played moves.

Note also that the functions $f, F_1, \ldots, F_n, \ldots$ suffice to recover the uniform strategy σ. This definition of uniform strategies is inspired by the work of Murawski and Ong [MO01].

Example 8: Let us go back to the play described in example 5. This play belongs to a strategy σ which is the interpretation of a $\lambda\mu 2$ term. As we shall prove further, this implies that σ is uniform.

We are interested by the two first moves of this play: $m_1 = \theta[a : H'/b_1; H/b_3]$

and $m_2 = [b : H'/b_2][g' : H''/b_4][h' :]$. The property of uniformity implies that:

- the choice of the nodes b, g', h' only depends on the choice of a by Opponent: $f(a) = a \cdot bg'h'$
- the closed arenas H' and H'' are determined by two arenas with holes $G_1 = F_1(a)$ and $G_2 = F_2(a)$, and we have $G_1[H'/C_1, H/C_2] = H'$ and $G_2[H'/C_1, H/C_2] = H''$; actually, G_1 is simply H' and $G_2 = C_2 \to \bot$.

The interest of this example lies in the comparison with the term $t = \Lambda Z.\Lambda X.\lambda x^{\forall Y.Y}.\lambda u^X.\lambda v^X.(x\{\forall U.U\}\{X \to \bot\})u$ interpreted by σ. Indeed, the arenas with holes G_1 and G_2 can in fact be read directly off this term: they correspond to the instantiations by $\forall U.U$ and $X \to \bot$. This case is actually a bit more simple than the definition, because each arena played by m_1 is of rank 1. ◇

The above definition of uniformity is not very convenient for its use in our proofs. That is why we introduce a lemma giving a very useful and widely used consequence of uniformity.

Lemma 10 *Let $s \in \sigma$ with σ uniform. Let $m \in s$ be such that $\lambda(m) = \mathbf{O}$ and H/b_i be one of its instantiations. Suppose that each node of H played during s is a root of H. Then, if H' is an arena whose roots have the same names as the roots of H, we have $s' \in \sigma$, where s' is the play s modified in the following way: first, the node m is replaced by m', which is identical to m except that it instantiates b_i by H'; second, each time an arena with holes does a reference to H, we give H' instead. We have, in particular, $\underline{s} = \underline{s'}$*

Proof: We set $s = s_1 m s_2$. By induction on the length of s_2:

- if $s_2 = m_1$, we have $\underline{s_1 m} = \underline{s_1 m'}$ and we can conclude thanks to uniformity
- if $s_2 = m_1 \dots m_n$ with $n \geq 3$, we have by induction hypothesis: $\underline{s_1 m m_1 \dots m_{n-2}} = \underline{s_1 m m'_1 \dots m'_{n-2}}$ (where m'_i is the move corresponding to m_i in s'); besides $m'_{n-1} = m_{n-1}$, so $\underline{s_1 m m_1 \dots m_{n-1}} = \underline{s_1 m m'_1 \dots m'_{n-1}}$, and we can conclude thanks to uniformity.

□

4.2 The uniform model

Our goal in this section is to prove that, by restricting the model to uniform strategies, we still obtain a control hyperdoctrine, whose structural objects are actually the same as in the original model. We first check that the basic

operations on strategies preserve uniformity:

Proposition 6 *If σ and τ are uniform and A is an arena, then $\sigma \times \tau$, $\sigma \parr A$, $A \parr \sigma$ and $\sigma;\tau$ are uniform.*

Proof: Preservation of uniformity through $\times$ is trivial. For $\parr$, we recall that $\sigma \parr A = \{s \in \mathcal{P}_{\vec{X}}((B \parr A) \to (C \parr A)) \mid s \restriction_{B\to C} \in \sigma \wedge s \restriction_{A\to A} \in id_A\}$: hence, if one can rebuild $\underline{sm \restriction_{B\to C}}$ from $\underline{s \restriction_{B\to C}}$ and $\overline{sm \restriction_{B\to C}}^{i}$ from $\overline{s \restriction_{B\to C}}^{i}$, one can rebuild $\underline{sm}$ from $\underline{s}$ and $\overline{\overline{sm}^{i}}$ from $\overline{s}^{i}$.

Let us focus our attention on composition.

Let $f, F_1, \ldots, F_n, \ldots$ and $f', F'_1, \ldots, F'_n, \ldots$ be the functions associated respectively with σ and τ. Let $smn \in \sigma;\tau$, we know that $smn = u \restriction_{A,C}$ with $u \in int(A,B,C)$, $u \restriction_{A,B} \in \sigma$ and $u \restriction_{B,C} \in \tau$. Besides, as the strategy is innocent, we can ask smn to be a thread. Finally, we note $s' = smn$.

Starting from a uniform strategy ρ, if we define $\underline{\rho} = \{\underline{s} \mid s \in \rho\}$, we can see that, thanks to uniformity, $\underline{\rho}$ is a strategy in a propositional setting. Thus we obtain two strategies $\underline{\sigma}$ and $\underline{\tau}$, which interact to give the play s' in particular. So we can use the Zipping lemma in the propositional setting (cf. [Har99]) to conclude that we can reconstruct $\underline{u}$ starting from $\underline{s'}$.

This gives us a partial function g_0 such that $\underline{u} = g_0(\underline{s'})$ (this function will be defined on s' if and only if we can reconstruct a finite sequence $\underline{u}$ starting from $\underline{s'}$). It is then easy to construct another partial function g so that $\underline{s'n} = g(\underline{s'})$.

Suppose now that n is in A (the case of n being in C can be treated exactly the same way), and note u' the prefix of u whose last move is m. We know that $\overline{n}^{i} = F_i(\underline{s_0})[\overline{s_0}^{i}]$, where $s_0 = u' \restriction_{A,B}$. As we can calculate $\underline{s_0}$ starting from $\underline{s'}$, this gives us $\overline{n}^{i} = H_i(\underline{s'})[\overline{s_0}^{i}]$. As s_0 is contained in u', we have $\overline{n}^{i} = H'_i(\underline{s'})[\overline{u'}^{i}]$. We will now prove in the following that we can keep this relation while suppressing from u' the moves played in B, one after the other.

Let m_0 be the last move in u' played in B and let us write $u' = s'_0 m_0 u_2$. m_0 is played by **P** for σ or for τ. Suppose it is for τ (the other case is equivalent): then we have $\overline{m_0}^{i} = F'_i(\underline{s'_0})[\overline{s'_0}^{i}]$, so $\overline{m_0}^{i} = F''_i(\underline{s'})[\overline{s'_0}^{i}]$ for some function F''_i. Note that formally $\overline{m_0}^{i}$ may differ if we consider σ or τ (because of the θ function), but it has no practical effect thanks to the definition of restriction (the θ functions are the same in plays of σ as in plays of τ). We now replace the sequence of arenas with holes $H_i(\underline{s'})$ by the adequate sequence of arenas with holes (i.e. where the arenas of $F''_i(\underline{s'})$ are placed in the corresponding

holes), and we get $\overline{n}^i = H''_i(\underline{s'})[\overline{s''_0}^i]$, where s''_0 is obtained by suppressing m_0 from u'.

We have succeeded in suppressing the last move of u' which is played in B, we can proceed iteratively until we only have moves played in A or C. This will give us a function G_i such that $\overline{n}^i = G_i(\underline{s'})[\overline{s'}^i]$. □

Proposition 7 *Let $\vec{X} = (X_1, \ldots, X_n)$ and $\vec{C} = (C_1, \ldots, C_n)$ where the free variables in $C_1, \ldots, C_n$ are chosen between $Y_1, \ldots, Y_m$. If $\sigma : A; \vec{X}$ is uniform then $\sigma[\vec{C}] : A[\vec{C}/\vec{X}]; \vec{Y}$ is uniform.*

Let $A, C \in \mathcal{G}(n+1)$, $D \in \mathcal{G}(n)$ and $\vec{X} = X_1, \ldots, X_n$. If $\sigma : F(\vec{X})(C) \to A; \vec{X}, X_{n+1}$ is uniform then $\kappa(\sigma) : C \to \forall X_{n+1}.A; \vec{X}$ is uniform. If $\tau : D \to \forall X_{n+1}.A; \vec{X}$ is uniform then $\kappa^{-1}(\tau) : D \to A; \vec{X}, X_{n+1}$ is uniform.

If $\sigma : A \to B; \vec{X}, X_{n+1}$ is uniform then $\forall\sigma : (\forall X_{n+1}.A) \to (\forall X_{n+1}.B); \vec{X}$ is uniform.

Proof: In each case, it suffices to do a slight update of the functions related to the first uniform strategy to obtain the new one. Note that this would not work for κ, κ^{-1} and $\sigma \mapsto \forall\sigma$ if we had not introduced the notion of rank: indeed, the rank is preserved through abstraction. □

It is easy to see that each base strategy is uniform:

Lemma 11 *If $A, B, C \in \mathcal{G}(k)$, then the following strategies are uniform: $\epsilon : \top$, $id_A : A \to A; \vec{X}$, $\Delta_A : A \to A \times A; \vec{X}$ and $\pi_A : A \times B \to A; \vec{X}$, $a_{A,B,C} : (A \parr B) \parr C \to A \parr (B \parr C)$, $l_A : A \to A \parr \bot$, $r_A : A \to \bot \parr A$, $i_A : \bot \to A$, $\nabla_A : A \parr A \to A$, $d_{A,B,C} : (A \parr C) \times (B \parr C) \to (A \times B) \parr C$, $s_{A,B,C} : (B \to A) \parr C \to ((B \parr C) \to A)$, as well as $a^{-1}_{A,B,C}$, l^{-1}_A, r^{-1}_A, $d^{-1}_{A,B,C}$ and $s^{-1}_{A,B,C}$.*

The control hyperdoctrine $\mathcal{M}$ was characterized by its base strategies, which are uniform, and its fundamental operations, which preserve uniformity. This leads us to the following:

Theorem 4 *If we restrain each category $\mathcal{G}(k)$ to the subcategory $\mathcal{G}_{unif}(k)$ where every strategy is uniform, we obtain a new control hyperdoctrine $\mathcal{M}_{unif}$.*

This model is much less symmetric and comfortable than the first one we introduced, that is why we introduced it only in a second time. But the symmetric model $\mathcal{M}$ is too liberal to allow us to deal properly with type isomorphisms; so, uniformity is an ad hoc property to constrain the model in such a way that we do not have more type isomorphisms in the model $\mathcal{M}_{unif}$ than in $\lambda\mu 2$, as we will see right now.

We are now ready to prove the fundamental result of our work on type isomorphisms: this is essentially the converse of the proposition 3. All this section is dedicated to the proof of this theorem.

Theorem 5 *If there exists a game isomorphism (σ, τ) between two polymorphic arenas A and B, with σ, τ uniform and innocent, then A and B are isomorphic.*

Proof: The same result has been proved in a propositional setting in [Lau05]; the tricky part in our case is that moves are not only nodes of a forest, but contain all the second-order structure. However, we may reuse the results of Olivier Laurent to do one part of the work.

Zig-zag plays:

Definition 33 (zig-zag play) *A play s of $A \to B$ is said to be **zig-zag** if*

- *each Player move following an Opponent move played in A (resp. in B) is played in B (resp. in A)*
- *each Player move played in A which follows an Opponent initial move played in B is justified by it*
- *$s \upharpoonright_A$ and $s \upharpoonright_B$ have the same pointers.*

If s is a zig-zag even-length play on $A \to B$, we note $\check{s}$ the unique zig-zag play on $B \to A$ such that $\check{s} \upharpoonright_A = s \upharpoonright_A$ and $\check{s} \upharpoonright_B = s \upharpoonright_B$.

We also give the definition of totality, which coincides with the usual notion:

Definition 34 (totality) *Let $\sigma : A; \vec{X}$. We say that σ is total if, whenever $s \in \sigma$ and $sm \in \mathcal{P}_{\vec{X}}(A)$, there exists a move m' such that $smm' \in \sigma$.*

Lemma 12 *If there is a game isomorphism (σ, τ) between A and B then:*

- *every play of σ or τ is zig-zag*
- *$\tau = \{\check{s} \mid s \in \sigma\}$*
- *σ and τ are total.*

This lemma can be proved by using forth and back translation to the propositional setting, as it has been defined in section 3.4. Indeed, as the lemma has been proved for the propositional setting in [Lau05], all we need is to check the following:

- if (σ, τ) defines an isomorphism between A and B then $([\![\sigma]\!]', [\![\tau]\!]')$ defines an isomorphism between $[\![A]\!]$ and $[\![B]\!]$
- if $[\![s]\!]'$ is zig-zag then s is zig-zag

- if $[\![\tau]\!] = \{\check{s} \mid s \in [\![\sigma]\!]\}$ then $\tau = \{\check{s} \mid s \in \sigma\}$
- if $[\![\sigma]\!]$ is total then σ is total

Given these preliminary results, we are ready to build the bijection $g : E_A \to E_B$ as a morphism for the whole structure of arenas: in order to do that, we will build g together with a bijection $\Psi : \mathcal{R}_A \to \mathcal{R}_B$ such that $g(\mathcal{T}(b)) = \mathcal{T}(\Psi(b))$ for all $b \in \mathcal{R}_A$, and then prove that $g(\mathcal{S}(b)) = \mathcal{S}(\Psi(b))$ and finally that $\mathcal{D}_A(c) = \mathcal{D}_B(g(c))$ for all $c \in E_A$.

Construction of the bijection g:

Suppose that the uniform strategies σ and τ are determined respectively by the functions $f, F_1, \ldots, F_n, \ldots$ and $f', F'_1, \ldots, F'_n, \ldots$. Consider an arena Q and a play (in a propositional setting) $s = m_1 \ldots m_n$ on the forest $\mathcal{F}_Q = (E_Q, \leq_Q)$; we can associate to this play a second-order play $[s] = [m_1] \ldots [m_n]$ on Q where each move $[m_i]$ consists in playing the node m_i (or the node replacing m_i) and instantiating each quantifier by $\perp$ (for an initial move, this also means that $\theta(X_j) = \perp$ for each X_j). This means that this move will be written $[m_i] = [m'_i : \perp/b_{i_1}; \ldots \perp/b_{i_n}]$ or $[m_i] = [m'_i : \perp/b_{i_1}; \ldots \perp/b_{i_n}][m''_i :]$ where m'_i and m''_i can be either m_i or a node of the form $m_i[\alpha'(x)/x]_{x \in V(m_i)}$, whose origin is m_i.

Let a be a node of A and $a_1 \ldots a_p$ be the sequence of nodes of A such that a_1 is a root, a_{i+1} son of a_i for $1 \leq i \leq p-1$ and $a_p = a$. By induction on p, we will define a function g from the nodes of A to the nodes of B and prove that [7] :

- if $s_\sigma = [g(a_1)][a_1][a_2][g(a_2)][g(a_3)][a_3] \ldots$ then $s_\sigma \in \sigma$
- there exists a bijection ψ between the quantifiers b_i such that $\mathcal{T}(b_i) = a_p$ and the quantifiers b'_i such that $\mathcal{T}(b'_i) = g(a_p)$, and we have : if s'_σ and s'_τ are respectively the plays s_σ and s_τ where the last move has been suppressed, $F_1(\underline{s'_\sigma}) = (C_{r(i)})_{i \in [1,n]}$ where $r(i)$ is the index associated with $b'_{\psi(i)}$ (or $b_{\psi^{-1}(i)}$), i.e. the index of the arena which instantiates this quantifier, and $F'_1(\underline{s'_\tau}) = (C_{l(i)})_{i \in [1,n]}$ where $l(i)$ is the index associated with $b_{\psi^{-1}(i)}$ (or $b'_{\psi(i)}$).

Note that the functions g and ψ should depend on a and be written g_a and ψ_a. But actually, by construction, we see that if a' is an ancestor of a, then $g_a a' = g_{a'}(a') = g(a')$, and the same holds for ψ.

If $p = 0$ it suffices to say that $\epsilon \in \sigma$ and $\epsilon \in \tau$.

[7] For a better understanding of this assertion, we recall that the arena C_i is the arena reduced to a node c_i.

If $p = p' + 1$ we note $s_1 = [a_1][g(a_1)][g(a_2)][a_2]...[g(a_{p'})][a_{p'}] \in \tau$
and $s_2 = [g(a_1)][a_1][a_2][g(a_2)]...[a_{p'}][g(a_{p'})] \in \sigma$. We choose the unique move m played in B such that $s_1[a_p]m \in \tau$ (it exists by totality of τ) and we set $g(a_p) = origin(m)$. Let $b_1, \dots, b_n$ be the quantifiers such that $\mathcal{T}(b_i) = a$ and $b'_1, \dots, b'_{n'}$ the quantifiers such that $\mathcal{T}(b'_i) = g(a_p)$.

By uniformity of τ, we have $\overline{m}^1 = F'_1(\underline{s_1[a_p]})[S_1]$ where S_1 stands for some substitution of holes and variables. As we have $s_2 m[a_p] \in \sigma$, we also get, from the uniformity of σ,

$$\overline{[a_p]}^1 = F_1(\underline{s_2 m})[S_2] \qquad (\star)$$

where S_2 stands for some substitution of holes and variables. We note $F_1(\underline{s_2 m}) = G_1 \dots G_n$ and $F'_1(\underline{s_1[a_p]}) = G'_1 \dots G'_n$, and we are interested in G_i for a given i. We know that $\overline{G_i[S_2]} = \bot$ from $(\star)$.

Let a' be the move obtained from $[a_p]$ by instantiating b_i by $H_1 = \neg H$ instead of $\bot$, where H is a non-empty closed arena. Then H_1 has the same root as $\bot$. We have $\underline{a'} = \underline{[a_p]}$, so $s_1 a' m' \in \tau$ and $s_2 m' a' \in \sigma$, with $\underline{m'} = \underline{m}$ thanks to lemma 10. This gives $\overline{a'}^1 = F_1(\underline{s_2 m'})[S'_2] = F_1(\underline{s_2 m})[S'_2]$ (S'_1 and S'_2 are the new substitutions obtained with the new instantiation). In particular, $H_1 = G_i[S'_2]$.

At this point, we have proved on one side $\bot = G_i[S_2]$ and on the other side $H_1 = G_i[S'_2]$. As $H_1 \neq \bot$ and S'_2 can only use [8] closed arenas from $\overline{s_2 m'}^1$ (and not from $\overline{s_2 m'}^2$). This means that G_i must contain a hole C_l referring to an arena from $\overline{m}^1$ (or $\overline{m'}^1$), because otherwise we should have $G_i[S'_2] = G_i[S_2]$ (indeed, the instantiation of b_i by H_1 instead of $\bot$ does not affect the closed arenas from $\overline{s_2}^1$, it affects only $\overline{m}^1$). But if, for each value l satisfying this property, G'_l does not contain the hole $C_{l(i)}$ (where $l(i)$ is the index of the arena which instantiates b_i in $s_1[a_p]m$), then $G'_l[S_1] = G'_l[S'_1]$, and so $G_i[S'_2] = G_i[S_2]$ (S_2 and S_2 can only be differentiated by the arenas $G'_l[S_1]$ and $G'_l[S'_1]$), which is absurd. We then have at least one value of l such that G_i contains the hole C_l and G'_l contains the hole $C_{l(i)}$.

It is then easy to see that $G'_l = C_{l(i)}$: indeed, if G'_l contained anything else than the node $c_{l(i)}$, $G'_l[S_1]$ would strictly contain $\bot$ (because none of the arenas it can refer to is empty), and $G_i[S_2] = \bot$ would also strictly contain $\bot$.

This proves that $n' \geq n$, and that we have an injection from the b_i quantifiers to the b'_j quantifiers. To show that $n \geq n'$, we will use nearly the same

[8] This is where it is important to use functions F_i indexed by the rank: if we could use some arenas from $\overline{s_2 m'}^2$ for example, then we could not conclude anything concerning the arenas appearing in $\overline{m'}^1$, and we could not prove that $n = n'$.

reasoning but in the other direction.

Let $j \in [1, n']$, we call K the closed arena by which m instantiates b'_j, i.e. $K = G'_j[S_1]$. Let m' be the move obtained from m by instantiating b'_j by K_1 instead of K, where K_1 must have the same roots as K but must be distinct from K (note that if K is empty we can choose $K_1 = \bot$). Then $\underline{m''} = \underline{m}$, so that we have $s_2 m'' a'' \in \sigma$ and $s_1 a'' m'' \in \tau$ with $\underline{a''} = \underline{[a_p]}$ thanks to lemma 10. So, $\overline{m''}^1 = F'_1(\underline{s_1 a''})[S''_1] = F'_1(\underline{s_1[a]})[S''_1]$ (S''_1 and S''_2 are the new substitutions obtained with the new instantiation). In particular, $K_1 = G'_j[S''_1]$.

At this point, we have on one side $K = G'_j[S_1]$ and on the other side $K_1 = G'_j[S''_1]$. We also know that $K \neq K_1$, and S''_1 can only use arenas from $\overline{s_1 a''}^1$, so G'_j contains at least one hole C_r which refers to an arena from $\overline{[a]}^1$ (or $\overline{a''}^1$) : indeed, the instantiation of b'_j by K_1 instead of K does not affect the closed arenas from $\overline{s_1}^1$, it only affects $\overline{[a]}^1$. But if, for any value of r satisfying this property, G_r does not contain the hole $C_{r(j)}$ (where $r(j)$ is the index of the arena which instantiates b'_j in $s_2 m[a_p]$), then $G_r[S''_2] = G_r[S_2]$, and so $G'_j[S''_1] = G'_j[S_1]$ which is absurd. We finally have at least one value of r such that G'_j contains the hole C_r and G_r contains the hole $C_{r(j)}$.

To see that $G_r = C_{r(j)}$, one must first define a move m_0 which is identical to m except that it does not instantiate any of its quantifiers by the empty arena (it puts $\bot$ instead, for example). This can modify K, but it does not modify the paths, so that, thanks to uniformity, we still have $s_2 m_0 a_0 \in \sigma$ and $s_1 a_0 m_0 \in \tau$, with $\underline{a_0} = \underline{[a]}$. Let S^0_1 and S^0_2 be the new substitutions obtained with the new instantiation. If G_r contained anything else than the hole $C_{r(j)}$, then $G_r[S^0_2]$ would strictly contain K (because none of the other arenas it can refer to is empty), and $G'_j[S^0_1] = K$ would strictly contain K.

Finally, we obtain a bijection ψ between the b_i's and the b'_j's ; m instantiates each of its quantifiers to $\bot$, so $m = [g(a_p)]$, and we have $F_1(\underline{s_2 m}) = (C_{r(i)})_{i \in [1,n]}$ where $r(i)$ is the index associated with $b'_{\psi(i)}$, and $F'_1(\underline{s_1[a_p]}) = (C_{l(i)})_{i \in [1,n]}$ where $l(i)$ is the index associated with $b_{\psi^{-1}(i)}$.

The case $p = p' + 1$ with p' odd can be treated exactly the same way, switching the roles of σ and τ.

It is now easy to associate a function g' to σ as we have associated a function g to τ. By construction, g and g' respect filiation (because of the property on zig-zag plays concerning pointers). We can easily check that $g \circ g'$ is the

identity on the nodes of A, and $g' \circ g$ is the identity on the nodes of B : this directly comes from the fact that $[g(a_1)][a_1][a_2][g(a_2)][g(a_3)][a_3]\cdots \in \sigma$ and $[a_1][g(a_1)][g(a_2)][a_2][a_3][g(a_3)]\cdots \in \tau$.

Proof of $g(\mathcal{R}_A) = \mathcal{R}_B$ and $\mathcal{D}_B \circ g = \mathcal{D}_A$:

The construction of the bijection $\Psi : \mathcal{R}_A \to \mathcal{R}_B$ such that $g(\mathcal{T}(b)) = \mathcal{T}(\Psi(b))$ for all $b \in \mathcal{R}_A$ is directly given by the function ψ. What remains to be proved is the following : if a_p appears n times in $\mathcal{S}(b_j)$ then $g(a_p)$ appears n times in $\mathcal{S}(\psi(b_j))$, and if a_p is decorated n times by X_k then $g(a_p)$ is also decorated n times by X_k.

Suppose that a_p appears n times in $\mathcal{S}(b_j)$ whereas $g(a_p)$ appears n' times in $\mathcal{S}(\psi(b_j))$ with for example $n' < n$. Let us consider the plays $s_1 = [a_1][g(a_1)]$ $[g(a_2)][a_2]...[g(a_p)][a_p] \in \tau$ and $s_2 = [g(a_1)][a_1][a_2][g(a_2)]...[a_p][g(a_p)] \in \sigma$ (here, p has been chosen even, but of course all this still holds for p odd), and take a play s' nearly identical to s_2 but for which b_j has been instantiated by the closed arena $H = \neg\neg \ldots \neg\bot$, which is a string of length N (i.e. a tree of depth N where each node has at most one son). Thanks to lemma 10, this play is still in σ. Besides, we note that, if $g(a_p) \in \mathcal{S}(b_r)$ (with $b_r \neq \psi(b_j)$) or if $g(a_p)$ is decorated by X_r, then b_r or X_r is still instantiated by $\bot$ in s' : indeed, the instantiation of b_r is given by the arena with holes $C_{\psi^{-1}(b_r)}$ (and $\psi^{-1}(b_r)$ is still instantiated by $\bot$), and the instantiation of X_r is given by a θ function which instantiates every variable by $\bot$. So, in the play s' the last move played is still $[g(a_p)]$, or $[g(a_p)]'$ ($[g(a_p)]'$ is just the move $[g(a_p)]$ where $\psi(b_j)$ has been instantiated by H: it is the case $\mathcal{T}(b_j) = a$) : we note this move $[g(a_p)]_0$. For simplicity of notations, we will consider from now that a_p does not appear in any other set of the form $\mathcal{S}(b_k)$, with $k \neq j$, and is not decorated by any variable X_k. The reader can easily check that this point does not change anything in the following, because each b_k, $k \neq j$, and each variable X_k, are instantiated by $\bot$.

Let us note $c_1, \ldots, c_N$ the successive nodes of the string H ($c_1 = c$). The play s' can be written $s' = t[a_p]_0[g(a_p)]_0$ ($[a_p]_0$ is either $[a_p]$, either the move which consists in playing $[a_p]$ by instantiating b_j by H, in the case $\mathcal{T}(b_j) = a$), and we have $\check{t}[g(a_p)]_0[a_p]_0 \in \tau$. The instantiations by H generate n strings on one side and n' strings on the other side: let us simply call the nodes of these strings $(a_p - c_2, i)$, $(a_p - c_3, i)$, $\ldots$, $(a_p - c_N, i)$, for $1 \leq i \leq n$, on one side, and $(g(a_p) - c_2, j)$, $(g(a_p) - c_3, j)$, $\ldots$, $(g(a_p) - c_N, j)$, for $1 \leq j \leq n'$, on the other side. Actually, these nodes correspond to "copies" of the nodes $c_1, \ldots, c_n$, and they are such that $|quant((a_p - c_j, i))| = |quant(c_j)| = 0$ (this can be proved for the substitution $D \mapsto D[H/b]$ by an induction on D).

For each $1 \leq i \leq n$, $\check{t}[g(a_p)]_0[a_p]_0(a_p - c_2, i) \in \mathcal{P}_{\vec{X}}(B \to A)$, so that we have $\check{t}[g(a_p)]_0[a_p]_0(a_p - c_2, i)m_i \in \tau$ for some move m_i, and m_i is justified by $[g(a_p)]_0$

(this is the property of zig-zag plays concerning pointers). Besides, the moves m_i must be pairwise distinct because $t[a_p]_0[g(a_p)]_0 m_i(a_p - c_2, i) \in \sigma$; but the node $g(a_p)$ has been substituted by $\mathfrak{P}_{j=1}^{n} H$, so there is one value for i such that the origin of m is a son of $g(a_p)$ *in the initial arena*. Indeed, there are only n' other alternatives, namely the moves $(g(a_p) - c_2, j)$ for $1 \leq j \leq n'$.

This implies that $g(a_p)$ has at least one son in the initial arena : the idea of the proof is to show that there is behind $g(a_p)$ a branch of length at least equal to N : this will lead us to a contradiction if we have chosen N big enough at the beginning (choose for example $n = h(A) + 1$, where $h(A)$ is the maximal depth of A).

As we have $\check{t}[g(a_p)]_0[a_p]_0(a_p - c_2, i)m \in \tau$, we also get $t[a_p]_0[g(a_p)]_0 m(a_p - c_2, i) \in \sigma$ and, as $t[a_p]_0[g(a_p)]_0 m(a_p c_2, i)(a_p c_3, i) \in \mathcal{P}_{\vec{X}}(A \to B)$, we have $t[a_p]_0[g(a_p)]_0$ $m(a_p c_2, i)(a_p - c_3, i)m' \in \sigma$ for some move m', justified by m and played in B. But be careful : this time, the origin of m, say d, may have been substituted by $\perp$ or by H, so the move m' is not necessarily played in the initial arena : it can be a move $(d - c_2, r)$, played in the arena substituted for d and corresponding to the node c_2 in H. In fact, we will show further that this case, which we call a **trapped substitution**, leads to a contradiction. Finally, one could think that an quantifier may have d as target, and this would lead to another substitution, by something else than $\perp$ or H ; but actually it suffices to reuse the arguments explained many times before to show, thanks to uniformity, that there cannot exist more quantifiers having d for target than having $(a_p - c_2, i)$ for target. As $|quant((a_p - c_2, i))| = 0$ this case is impossible.

We can thus go on with our proof : $origin(m')$ is a son of $origin(m)$. We have $\check{t}[g(a_p)]_0[a_p]_0(a_p - c_2, i)mm'(a_p c_3, i) \in \tau$, and $\check{t}[g(a_p)]_0[a_p]_0(a_p - c_2, i)mm'(a_p c_3, i)(a_p - c_4, i) \in \mathcal{P}_{\vec{X}}(B \to A)$, so $\check{t}[g(a_p)]_0[a_p]_0(a_p - c_2, i)mm'(a_p - c_3, i)(a_p - c_4, i)m'' \in \tau$ for some move m'' justified by m', etc. So, by systematically rejecting trapped substitutions, we show that we can construct a branch of length N descending from $g(a_p)$. This leads to a contradiction, if we have chosen N big enough at the beginning.

We still have to show the impossibility of trapped substitutions : suppose that $t[a_p]_0[g(a_p)]_0 m_0(a_p - c_2, i)(a_p, c_3 - i)m_1 m_2 \ldots (a_p - c_k, i)(a_p - c_{k+1}, i)m_l \in \sigma$ (or, equivalently, $\check{t}[g(a_p)]_0[a_p]_0(a_p - c_2, i)m_0 m_1(a_p - c_3, i)(a_p - c_4, i) \ldots (a_p - c_k, i)(a_p - c_{k+1}, i)m_l \in \tau$) with $k \geq 2$ and $m_l = (d - c_2, r)$ where $d = \text{origin}(m_{l-1})$. This would correspond to the case where one of the descendants of $g(a_p)$ belongs to $\mathcal{S}(\psi(b_j))$, and we have then played in the substituted arena. In this case, we have $t[a_p]_0[g(a_p)]_0 m_0(a_p - c_2, i)(a_p - c_3, i)m_1 m_2 \ldots (a_p - c_k, i)(a_p - c_{k+1}, i)m_l(d - c_3, r) \in \mathcal{P}_{\vec{X}}(A \to B)$, so $t[a_p]_0[g(a_p)]_0 m_0(a_p - c_2, i)(a_p - c_3, i)m_1 m_2 \ldots (a_p - c_k, i)(a_p - c_{k+1}, i)m_l(d - c_3, r)(a_p - c_{k+2}, i) \in \sigma$ by totality (because $(a_p - c_{k+2}, i)$, if it exists, is the only son of $(a_p - c_{k+1}, i)$). So $\check{t}[g(a_p)]_0[a_p]_0(a_p - c_2, i)m_0 m_1(a_p - c_3, i)(a_p - c_4, i) \ldots (a_p - c_{k+2}, i)(d - c_3, r) \in \tau$, and, by the same arguments, $\check{t}[g(a_p)]_0[a_p]_0(a_p -$

$c_2, i)m_0m_1(a_p - c_3, i)(a_p - c_4, i)\ldots(a_p - c_{k+2}, i)(d - c_3, r)(d - c_4, r)(a_p - c_{k+3}, i) \in \tau$, etc. Finally, as $k \geq 2$, we necessarily come to a contradiction, namely looking for a son of c_N, which does not have any by construction.

We have proved that if a_p appears n times in $\mathcal{S}(b_j)$ then $g(a_p)$ appears n times in $\in \mathcal{S}(\psi(b_j))$. We still should show that if a_p is decorated n times by X_j then $g(a_p)$ is decorated n times by X_j. But actually we see immediately that the arguments are exactly the same : here, it suffices to consider the plays s_1 and s_2 with a new function θ' which instantiates each variable by $\bot$, except X_j which is instantiated by H. By lemma 10 we still obtain a play from σ or τ, and we can follow the preceding proof without any trouble. □

4.4 Isomorphisms of types

Thanks to the preceding result, and to the fact that we have a model of $\lambda\mu 2$, we are able to characterize precisely type isomorphisms in this system. The equational system we want to establish for type isomorphisms in $\lambda\mu 2$ has been presented on figure 4.

On the grammar of $\lambda\mu 2$ types, we consider:

- products of arity n: $\prod_{i=1}^{n} M_i = ((M_1 \times M_2) \times \ldots) \times M_n$ ($\prod_{i=1}^{n} M_i = \top$ if $n = 0$)
- disjunctions of arity n: $\parr_{i=1}^{n} M_i = ((M_1 \parr M_2) \parr \ldots) \parr M_n$ ($\parr_{i=1}^{n} M_i = \bot$ if $n = 0$)
- quantifications of arity n: $\overrightarrow{\forall X}_M = \forall X_{i_1} \ldots \forall X_{i_n}$ if $M = \{i_1, \ldots, i_n\}$.

Inspired by the work of Roberto Di Cosmo on system F types[DC95], we define normal forms:

Definition 35 (canonical form) *A second order type N is called a **canonical form** if it is written $N = \prod_{i=1}^{n} \overrightarrow{\forall X}_{M_i}.N_i \to \alpha_i$ with $\alpha_i = \parr_{j=1}^{m} X_{k_j}$ and N_i canonical form.*

Lemma 13 *Let A be a type in $\lambda\mu 2$. There exists a canonical form A' such that $A \simeq_\varepsilon A'$.*

Proof : Because of the associativity of $\times$, $\parr$ and $\forall$ in $\simeq_\varepsilon$, we can restrict ourselves to products, disjunction and quantifications of arity n. Then, modulo α-equivalence, canonical forms are the normal forms of the following rewriting system:

$$(A \times B) \parr C \Rightarrow (A \parr B) \times (B \parr C) \qquad A \parr \bot \Rightarrow A$$
$$(A \to B) \parr C \Rightarrow A \to (B \parr C) \qquad \bot \parr A \Rightarrow A$$
$$A \to (B \times C) \Rightarrow (A \to B) \times (A \to C) \qquad \top \parr A \Rightarrow \top$$
$$A \to (B \to C) \Rightarrow (A \times B) \to C \qquad A \parr \top \Rightarrow \top$$
$$(\forall X.A) \parr B \Rightarrow \forall X.(A \parr B) \qquad A \times \top \Rightarrow A$$
$$\forall X.(A \times B) \Rightarrow (\forall X.A) \times (\forall X.B) \qquad \top \times A \Rightarrow A$$
$$A \to \forall X.B \Rightarrow \forall X.(A \to B) \qquad A \to \top \Rightarrow \top$$
$$\forall X.\top \Rightarrow \top$$

This rewriting system is coherent with $\simeq_\varepsilon$: this means that if $A \Rightarrow A'$ then $A \simeq_\varepsilon A'$. To show that this system terminates, we define a function ψ which associates to each second order type A a natural number $\psi(A) \geq 2$:

$$\psi(A \times B) = \psi(A) + \psi(B) + 1$$
$$\psi(\forall X.A) = 2\psi(A)$$
$$\psi(A \to B) = \psi(A)\psi(B) + 1$$
$$\psi(A \parr B) = 2^{\psi(A)\psi(B)}$$
$$\psi(\top) = \psi(\bot) = \psi(Y) = 2$$

where Y stands for any type variable.

For each rewriting rule $A \Rightarrow A'$, we have $\psi(A) > \psi(A')$. □

Proposition 8 *If A and B are two types built on the grammar of $\lambda\mu 2$ such that A^* and B^* are isomorphic, then $A \simeq_\varepsilon B$.*

Proof: In this proof we are interested by the hyperforest structure of A^* and B^* rather than their arborescence.

Let g and ψ be the bijections which characterize the isomorphism between A^* and B^*. Suppose that A and B are already in a canonical form, we will show that these two forms are equal modulo $\simeq_\varepsilon$ by induction on the structure of A^*:

- If A^* is empty, then B^* is empty and $A^* \simeq_\varepsilon B^*$.
- If A^* is a tree such that no hyperedge has the root as target, then B^* is a tree such that no hyperedge has the root as target. Then $A \simeq_\varepsilon A' \to (X_{i_1} \parr \dots X_{i_n})$ (this is indeed the only normal form which can be interpreted by such a tree) with X_i free type variable and we have in

this case, because of the bijection g which sends $\mathcal{D}_A$ to $\mathcal{D}_B$, $B \simeq_\varepsilon B' \to (X_{i_{\sigma(1)}} \parr \dots X_{i_{\sigma(n)}})$ where σ is a permutation of $\{1, \dots, n\}$. We obtain $(A')^*$ (resp. $(B')^*$) by suppressing the root from A^* (resp from B^*), so $(A')^*$ and $(B')^*$ are isomorphic. Finally, by induction hypothesis, $A' \simeq_\varepsilon B'$, so $A \simeq_\varepsilon B$.

- if A^* is a tree (whose root is denoted r) with some hyperedges $b_1, \dots, b_n$ such that $\mathcal{T}(b_i) = r$ for $i \in [1, n]$, then B^* is also a tree (whose root is denoted r') with some hyperedges $b'_1, \dots, b'_n$ such that $\mathcal{T}(b'_j) = r$ for $j \in [1, n]$. Then we have $A \simeq_\varepsilon \forall X_1 \dots \forall X_n . A'$ (this is the only possible representation by a normal form) and $B \simeq_\varepsilon \forall X'_1 \dots \forall X'_n . B'$, where each X_k is associated with some b_i and each X'_k is associated with some b'_j (the variables X_i and X'_i are chosen fresh with respect to other free variables already occurring in A and B). By α-renaming, we can choose the variables X'_k such that: if X_k is the variable associated to the hyperedge b_i, then the variable associated to $\psi(b_i)$ is X_k. $(A')^*$ (resp. $(B')^*$) is obtained from A^* (resp. B^*) by suppressing all hyperedges $b_1, \dots, b_n$ (resp. $b'_1, \dots, b'_n$) and by decorating with X_i each node c such that $c \in \mathcal{S}(b_i)$ (resp. $c \in \mathcal{S}(b'_i)$). By using the property $\mathcal{S} \circ \psi = g \circ \mathcal{S}$, we see that $(A')^*$ and $(B')^*$ are isomorphic, so $A' \simeq_\varepsilon B'$ and finally $A \simeq_\varepsilon B$ by commutativity of quantifications.
- If A^* contains $k \geq 2$ trees, then B^* also contains $k \geq 2$ trees and A is obtained from k formulas $A_1, \dots, A_k$ by using the connector $\times$, so by associativity $A \simeq_\varepsilon ((A_1 \times A_2) \times A_{k-1}) \times A_k$, where each A^*_i is a tree of A^* (this is indeed the only representation by a normal form). In the same way, we have $\simeq_\varepsilon ((B_1 \times B_2) \times B_{k-1}) \times B_k$ where each $[\![B_i]\!]$ is a tree of $[\![B]\!]$. As A^* and B^* are isomorphic, one can find a permutation ϕ of the trees of A^* such that, for every $1 \leq i \leq k$, $A^*_{\phi(i)}$ and B^*_i are isomorphic. By induction hypothesis, this implies $A_{\phi(i)} \simeq_\varepsilon B_i$, so by commutativity of $\times$ we have $A \simeq_\varepsilon B$.

□

Theorem 6 *Two formulas A and B are isomorphic in $\lambda\mu 2$ if and only if $A \simeq_\varepsilon B$ (with $\simeq_\varepsilon$ defined on figure 2 p. 4).*

Proof: If $A \simeq_\varepsilon B$ then A and B are isomorphic in the $\lambda\mu 2$-calculus: to prove it we just have to give a couple of terms realizing the isomorphism. As an example we give the isomorphism between $A \parr (B \parr C)$ and $(A \parr B) \parr C$:

$$\begin{cases} \vdash t : A \parr (B \parr C) \to (A \parr B) \parr C \\ \vdash u : (A \parr B) \parr C \to A \parr (B \parr C) \end{cases}$$

with

$$t = \lambda x^{A \parr (B \parr C)} . \mu(\alpha_2^{A \parr B}, \beta_1^C) . [\alpha_2] \mu(\alpha_0^A, \alpha_1^B) . [\alpha_1, \beta_1] \mu \beta_0^{B \parr C} . [\alpha_0, \beta_0] x$$

and

$$u = \lambda x^{(A \parr B) \parr C}.\mu(\alpha_1^A, \alpha_2^{B \parr C}).[\alpha_2]\mu(\beta_1^B, \beta_0^C).[\alpha_1, \beta^1]\mu\alpha_0^{A \parr B}.[\alpha_0, \beta_0]x$$

For the other implication, suppose there are two terms $u : A \to B$ and $v : B \to A$ such that $u \circ v = id_B$ and $v \circ u = id_A$. In the uniform model, their respective interpretations σ_u and σ_v are such that $\sigma_v; \sigma_u = id_B$ and $\sigma_u; \sigma_v = id_A$. We then have a game isomorphism between the arenas A^* and B^*, so A^* and B^* are isomorphic, so that $A \simeq_\varepsilon B$. □

Corollary 1 *If we consider the system $\lambda\mu 2'$ obtained by suppressing the constructors $[\alpha, \beta]t$ and $\mu(\alpha^A, \beta^B).t$ from the grammar of terms (as well as their associated inference rules and reduction rules) and the constructor $\parr$ from the grammar of types, then type isomorphisms in $\lambda\mu 2'$ are characterized by the equational system $\simeq'_\varepsilon$ given on figure 4.*

$A \times B \simeq'_\varepsilon B \times A$	$A \times \top \simeq'_\varepsilon A$	$\forall X.\forall Y.A \simeq'_\varepsilon \forall Y.\forall X.A$
$A \times (B \times C) \simeq'_\varepsilon (A \times B) \times C$	$\forall X.\top \simeq'_\varepsilon \top$	$\forall X.(A \times B) \simeq'_\varepsilon \forall X.A \times \forall X.B$
$A \to (B \to C) \simeq'_\varepsilon (A \times B) \to C$	$\top \to A \simeq'_\varepsilon A$	$A \to \forall X.B \simeq'_\varepsilon \forall X.(A \to B)$
$A \to (B \times C) \simeq'_\varepsilon (A \to B) \times (A \to C)$	$A \to \top \simeq'_\varepsilon \top$	if X does not appear free in A

Fig. 4. Equational system for type isomorphisms in $\lambda\mu 2'$ and in system F

Proof: As this new system is included in $\lambda\mu 2$, our model is necessarily also a model of $\lambda\mu 2'$. Thanks to the theorem 5, we only need to check that: if A and B are two types built on the grammar of $\lambda\mu 2'$ such that A^* and B^* are isomorphic, then $A \simeq'_\varepsilon B$ (and also that there exist terms in $\lambda\mu 2'$ realizing the equations of $\simeq'_\varepsilon$). □

Finally, we also recover the results of Roberto Di Cosmo [DC95]:

Corollary 2 *Type isomorphisms for system F are given by the equational system $\simeq'_\varepsilon$.*

Proof: System F is $\lambda\mu 2'$ where we have suppressed the constructors $[\alpha]t$ and $\mu\alpha^A.t$. The type system is the same as the one of $\lambda\mu 2'$, so we conclude by the same reasoning as above. □

5 Conclusion and further directions

Game semantics has allowed us not only to retrieve semantically the results of Robert Di Cosmo concerning type isomorphisms in system F, but also to give a characterization of type isomorphisms for an extension of this system to a calculus with control.

However, although it has led us to a good characterization of type isomorphisms, uniformity is still a very ad hoc property. We suspect that there is an analogy between uniformity and innocence: maybe this link could help us understanding uniformity better.

Concerning extension, one can easily adapt our model to a calculus with a fixpoint operator: it only requires to enrich the structure of the model with an complete partial order on strategies (which will simply be the inclusion of strategies). Note that it is possible only because we did not require the totality of strategies in our model (as we were able to prove the totality of strategies realizing game isomorphisms). Moreover, our strategies are considered here as innocent, but in fact it would suffice to have them well-threaded and visible: in this case, our model would appear to be a good candidate to interpret second-order Idealized Algol [AM99] and to characterize type isomorphisms in this system by a similar equational system. Many other programming features may also be treated using this kind of model, like non-determinism, probabilities, concurrency, ... Taking the same approach for Curry-style system F or for ML should also be possible, but it will require to build a brand new model, because the model we give here does not suit these systems.

Finally, our approach needs to be tested on retractions, i.e. in the case where we have $f : A \to B$ and $g : B \to A$ such that $f \circ g = id_B$ but not necessary $g \circ f = id_A$. Retractions can be understood as a subtyping notion, and they are useful when dealing with code reuse (because, schematically, it is no big deal to use a function having a "more liberal" type than the one we expected). In this domain few results [dLPS92, Pad01, RU02] are known, even in a propositional setting, so game semantics may give a new enlightenment on this problem.

References

[AJ03] Samson Abramsky and Radha Jagadeesan. A game semantics for generic polymorphism. In Andrew D. Gordon, editor, *Foundations of Software Science and Computational Structures*, volume 2620 of *LNCS*, pages 1–22. Springer, 2003.

[AJM00] Samson Abramsky, Radha Jagadeesan, and Pasquale Malacaria. Full abstraction for PCF. *Information and Computation*, 163(2):409–470, December 2000.

[AM99] Samson Abramsky and Guy McCusker. Full abstraction for idealized algol with passive expressions. *Theoretical Computer Science*, 227:3–42, September 1999.

[BP01] Gilles Barthe and Olivier Pons. Type isomorphisms and proof reuse in dependent type theory. In F. Honsell and M. Miculan, editors, *Foundations of Software Science and Computation Structures*, volume 2030 of *LNCS*, 2001.

[DC95] Roberto Di Cosmo. *Isomorphisms of Types*. Progress in Theoretical Computer Science. Birkhäuser, 1995.

[dLPS92] Ugo de' Liguoro, Adolfo Piperno, and Richard Statman. Retracts in simply typed $\lambda\beta\eta$-calculus. In *Proceedings of the eleventh annual symposium on Logic In Computer Science*, pages 461–469. IEEE, IEEE Computer Society Press, 1992.

[Har99] Russel Harmer. *Games and Full Abstraction for Nondeterministic Languages*. Ph.D. thesis, Imperial College and University of London, 1999.

[HO00] Martin Hyland and Luke Ong. On full abstraction for PCF. *Information and Computation*, 163(2):285–408, December 2000.

[Hug97] Dominic Hughes. Games and definability for system F. In *Logic in Computer Science*. IEEE, 1997.

[Hug00] Dominic Hughes. *Hypergame semantics: full completeness for system F*. D.Phil. thesis, Oxford University, 2000.

[Lai97] James Laird. Full abstraction for functional languages with control. In *Proceedings of the twelfth annual symposium on Logic In Computer Science*, pages 58–67, Warsaw, June 1997. IEEE, IEEE Computer Society Press.

[Lau02] Olivier Laurent. *Étude de la polarisation en logique*. Thèse de doctorat, Université Aix-Marseille II, March 2002.

[Lau05] Olivier Laurent. Classical isomorphisms of types. *Mathematical Structures in Computer Science*, 15(5):969–1004, October 2005.

[Law70] F. W. Lawvere. Equality in hyperdoctrines and the comprehension schema as an adjoint functor. In *Proceedings on Applications of Categorical Logic*, 1970.

[MO01] Andrzej Murawski and Luke Ong. Evolving games and essential nets for affine polymorphism. In Samson Abramsky, editor, *Typed Lambda Calculi and Applications '01*, volume 2044 of *LNCS*. Springer, 2001.

[Nic94] Hanno Nickau. Hereditarily sequential functionals. In Anil Nerode and Yuri Matiyasevich, editors, *Logical Foundations of Computer Science*, volume 813 of *LNCS*, pages 253–264. Springer, 1994.

[Pad01] Vincent Padovani. Retracts in simple types. In Samson Abramsky, editor, *Typed Lambda Calculi and Applications '01*, volume 2044 of

LNCS, pages 376–384. Springer, May 2001.

[Par92] Michel Parigot. $\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction. In *Proceedings of International Conference on Logic Programming and Automated Reasoning*, volume 624 of *LNCS*, 1992.

[Pit88] Andrew Pitts. Polymorphism is set-theoretic constructively. In D. Pitt, editor, *CTCS*, volume 283 of *LNCS*, 1988.

[Rit91] Mikael Rittri. Using types as search keys in function libraries. *Journal of Functional Programming*, 1(1):71–89, 1991.

[RU02] Laurent Regnier and Pawel Urzyczyn. Retractions of types with many atoms. *CoRR*, cs.LO/0212005, 2002.

[See87] R. A. G. Seely. Categorical semantics for higher-order polymorphic lambda-calculus. *Journal of Symbolic Logic*, 52(4):969–989, 1987.

[Sel01] Peter Selinger. Control categories and duality: on the categorical semantics of the lambda-mu calculus. *Mathematical Structures in Computer Science*, 11:207–260, 2001.

[Sol83] Sergei Soloviev. The category of finite sets and cartesian closed categories. *Journal of Soviet Mathematics*, 22(3):1387–1400, 1983.

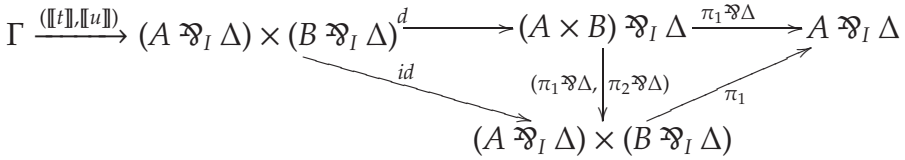
A Soundness of the interpretation of $\lambda\mu 2$ in a control hyperdoctrine

Theorem 1 (soundness) *The interpretation of second-order $\lambda\mu$-terms in a control hyperdoctrine is sound: for any couple of terms t, u such that $t = u$, we have $[\![t]\!] = [\![u]\!]$.*

Proof: We prove successively the soundness of the interpretation for every reduction rule. Most of the required equalities are already valid (with the same proof) in a control category or in a hyperdoctrine. That is why, in many cases, we will only give a sketch of the proof: we do not detail the commutativity of the diagrams and the validity of substitution lemmas.

($\top$) It suffices to recall that $1 \parr_I \Delta$ is isomorphic to 1, and 1 is a terminal object.

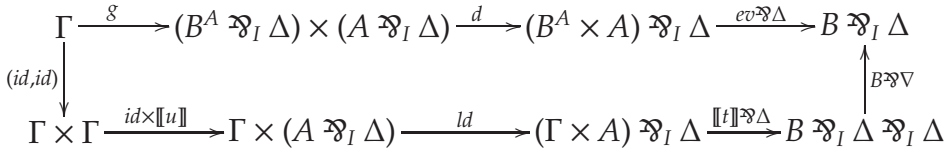
(π_i) As $d^{-1}_{A,B,\Delta} = (\pi_1 \parr_I \Delta, \pi_2 \parr_I \Delta)$, one has:



($\times$) This comes directly from $d^{-1}_{A,B,\Delta} = (\pi_1 \parr_I \Delta, \pi_2 \parr_I \Delta)$.

(β) Let us introduce the **linear distributivity** $ld : A \times (B \parr_I C) \xrightarrow{w\times id} (A \parr_I C) \times (B \parr_I C) \xrightarrow{d} (A \times B) \parr_I C$.

The first step is to prove that the following diagram commutes:



with $g = (\Lambda([\![t]\!]); s^{-1}, [\![u]\!])$.

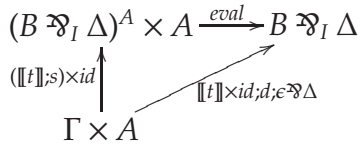
Then, one has to prove the following substitution lemma (by induction on t):

$$(id, id); id \times [\![u]\!]; ld; [\![t]\!] \parr_I \Delta; id \parr_I \Delta = [\![t[u/x]]\!]$$

(η) What we want to show is:

$$\Lambda(([\![t]\!], \pi_1); d; \epsilon \parr_I \Delta); s^{-1} = [\![t]\!]$$

And this precisely means:



which is straightforward.

(μ) For ($\mu^{\to}$), the substitution lemma to prove by induction on t is (modulo some trivial morphisms):

$$id \times [\![u]\!]; [\![t]\!] \times id; d; \epsilon \parr \Delta = [\![t[[\beta](-)u/[\alpha](-)]]\!]$$

The most significant case of the induction is the case where $t = [\alpha]t'$: it consists in this case in proving that (informally) $\nabla; (d; \epsilon \parr \Delta) = (d; \epsilon \parr \Delta); \nabla$: it can be done using curryfication and decurryfication (using the fact that B^v is central for any morphism v).

For ($\mu^{\forall}$), first remark that:

$$(\kappa^{-1}([\![t]\!]; p))[U^n, B] = ([\![t]\!])^{U^{n+1}}_{U^n}[U^n, B]; \kappa^{-1}(p)[U^n, B]$$

Then the substitution lemma to prove by induction on t is (modulo some trivial morphisms):

$$[\![t]\!]; \kappa^{-1}(p)[U^n, B] = [\![t[[\beta](-)\{B\}/[\alpha](-)]]\!]$$

The most significant case of the induction is the case where $t = [\alpha]t'$: it consists in this case in proving that (informally) $\nabla; \kappa^{-1}(p)[U^n, B] = \kappa^{-1}(p)[U^n, B] \parr \kappa^{-1}(p)[U^n, B]; \nabla$: this is ensured by focality of $\kappa^{-1}(p)[U^n, B]$ (which is due to the centrality of this morphism).

The rules ($\mu^{\times}$) and ($\mu^{\parr}$) can be treated similarly.

(ρ) The three rules (ρ^{μ}), ($\rho^{\parr}$) and ($\rho^{\bot}$) work on the same scheme. If we look for example to (ρ^{μ}), one has:

$$\begin{aligned}
[\![[\alpha']\mu\alpha^A.t]\!] &= \Gamma \xrightarrow{[\![t]\!]} \bot_I \parr_I A \parr_I \Delta \xrightarrow{\cong} A \parr_I \Delta \xrightarrow{w \parr \Delta} \Delta \parr_I \Delta \xrightarrow{\nabla} \Delta \xrightarrow{\cong} \bot_I \parr_I \Delta \\
&= \Gamma \xrightarrow{[\![t]\!]} \bot_I \parr_I A \parr_I \Delta \xrightarrow{c;\Delta} \bot_I \parr_I \Delta \\
&= [\![t[\alpha'/\alpha]]\!]
\end{aligned}$$

(θ) If we focus on (θ^{μ}) for example, note that the derivations we want to compare are the following ones:

$$\frac{\sigma}{\vec{X}, \Gamma \vdash t : A \mid \alpha : A, \Delta} \qquad \frac{\dfrac{\dfrac{W(\sigma)}{\vec{X}, \Gamma \vdash t : A \mid \alpha : A, \alpha' : A, \Delta}}{\vec{X}, \Gamma \vdash [\alpha]t : \bot \mid \alpha : A, \alpha' : A, \Delta}}{\vec{X}, \Gamma \vdash \mu\alpha^A.[\alpha]t : A \mid \alpha' : A, \Delta}$$

where $W(\sigma)$ is obtained from σ by a weakening lemma.

Hence, we have to compare

$$\Gamma \xrightarrow{[\![t]\!]} A \parr_I A \parr_I \Delta$$

with

$$[\![\mu\alpha^A.[\alpha]t]\!] = \Gamma \xrightarrow{[\![W(t)]\!]} A \parr_I (A \parr_I A \parr_I \Delta) \xrightarrow{w} (A \parr_I A \parr_I \Delta) \parr_I (A \parr_I A \parr_I \Delta) \xrightarrow{\nabla} A \parr_I A \parr_I \Delta$$

By induction on the proof σ, one can show that actually $[\![W(t)]\!] = \Gamma \xrightarrow{[\![t]\!]} A \parr_I (A \parr_I \Delta) \xrightarrow{w} A \parr_I (A \parr_I A \parr_I \Delta)$. Besides, as $\alpha \notin FN(t)$, one can also prove by induction on σ that $[\![t]\!] = \Gamma \xrightarrow{[\![t']\!]} A \parr_I \Delta \xrightarrow{w} A \parr_I (A \parr_I \Delta)$ for some t'.

Hence, we have two weakening composing with contractions that give us the identity, and $[\![t']\!]$ is composed with the last weakening to give us $[\![t]\!]$.

(β2) $[\![\Lambda X.t]\!]\{B\} = [\![t]\!][U^n, B]$, so one only needs to prove a substitution lemma:

$$[\![t]\!][U^n, B] = [\![t[B/X]]\!]$$

which is ensured by the fact that specialization functors are strict functors of control categories (indeed, they are strict functors of pre-control categories, and the strictness for other structural morphisms is automatically true).

(η2) This is immediate, since:

$$\kappa((\kappa^{-1}([\![t]\!]; p))[U^n, U]); p^{-1} = [\![t]\!]$$